PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : G06F 17/60 — A2

(11) International Publication Number: WO 95/26534

(43) International Publication Date: 5 October 1995 (05.10.95)

(21) International Application Number: PCT/CA95/00166

(22) International Filing Date: 23 March 1995 (23.03.95)

(30) Priority Data:
08/217,913 25 March 1994 (25.03.94) US

(60) Parent Application or Grant
(63) Related by Continuation
US 08/217,913 (CON)
Filed on 25 March 1994 (25.03.94)

(71) Applicant *(for all designated States except US)*: INTERNATIONAL TRADE DIRECTORIES INC. [CA/CA]; RR#4, P.O. Box 20025, Picton, Ontario K0K 3V0 (CA).

(72) Inventors; and
(75) Inventors/Applicants *(for US only)*: PORTELANCE, Diane [CA/CA]; R.R. #4 Glenora Road, Picton, Ontario K0K 2T0 (CA). BENNETT, Carl, Austin [CA/CA]; Box 21024, Kingston, Ontario K7L 5P5 (CA).

(74) Agent: BERESKIN & PARR; 40th floor, 40 King Street West, Toronto, Ontario M5H 3Y2 (CA).

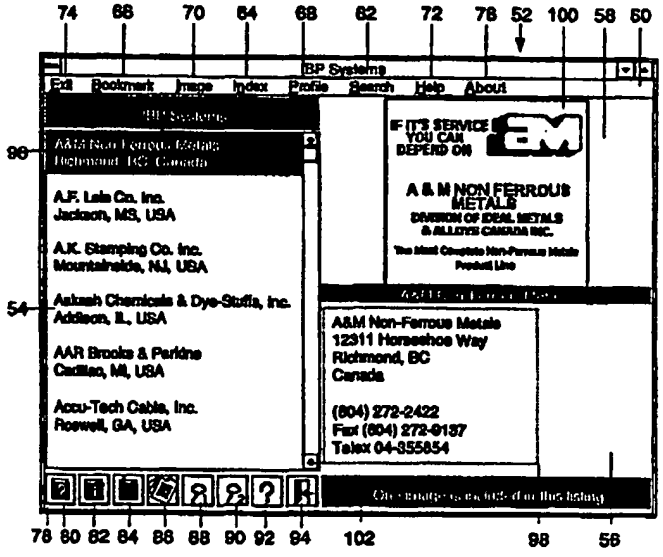
(81) Designated States: AM, AT, AU, BB, BG, BR, BY, CA, CH, CN, CZ, DE, DK, EE, ES, FI, GB, GE, HU, IS, JP, KE, KG, KP, KR, KZ, LK, LR, LT, LU, LV, MD, MG, MN, MW, MX, NL, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, TJ, TT, UA, UG, US, UZ, VN, European patent (AT, BE, CH, DE, DK, ES, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, ML, MR, NE, SN, TD, TG), ARIPO patent (KE, MW, SD, SZ, UG).

Published
Without international search report and to be republished upon receipt of that report.

(54) Title: GRAPHIC TRADE DIRECTORY WITH DATA COMPRESSION

(57) Abstract

A computer trade directory with subscriber advertising and data and image compression. The trade directory comprises a graphical user interface and a series of data volumes containing company listing information and images. The graphical user interface provides search and index functions allowing a user to locate company listing information by company name, city, province/state, country, product classification and sub-classification. Images containing advertising are automatically displayed when searching for suppliers of items in a particular product classification. The display mode and prominence of an advertising image can be controlled according to a subscriber's preference. The invention includes compression algorithms for encoding and storing company listing information and images which can comprise a mix of text, line drawings, photographs, spot colour and true colour. The trade directory can be installed in compressed form on the hard disk of a computer or run directly from floppy diskettes. In operation, company listing information, including images, are selectively unpacked from disk as needed. This eliminates the need for CD-ROM devices and allows the trade directory to be distributed to a wide range of users.



FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AT	Austria	GB	United Kingdom	MR	Mauritania
AU	Australia	GE	Georgia	MW	Malawi
BB	Barbados	GN	Guinea	NE	Niger
BE	Belgium	GR	Greece	NL	Netherlands
BF	Burkina Faso	HU	Hungary	NO	Norway
BG	Bulgaria	IE	Ireland	NZ	New Zealand
BJ	Benin	IT	Italy	PL	Poland
BR	Brazil	JP	Japan	PT	Portugal
BY	Belarus	KE	Kenya	RO	Romania
CA	Canada	KG	Kyrgystan	RU	Russian Federation
CF	Central African Republic	KP	Democratic People's Republic of Korea	SD	Sudan
CG	Congo			SE	Sweden
CH	Switzerland	KR	Republic of Korea	SI	Slovenia
CI	Côte d'Ivoire	KZ	Kazakhstan	SK	Slovakia
CM	Cameroon	LI	Liechtenstein	SN	Senegal
CN	China	LK	Sri Lanka	TD	Chad
CS	Czechoslovakia	LU	Luxembourg	TG	Togo
CZ	Czech Republic	LV	Latvia	TJ	Tajikistan
DE	Germany	MC	Monaco	TT	Trinidad and Tobago
DK	Denmark	MD	Republic of Moldova	UA	Ukraine
ES	Spain	MG	Madagascar	US	United States of America
FI	Finland	ML	Mali	UZ	Uzbekistan
FR	France	MN	Mongolia	VN	Viet Nam
GA	Gabon				

GRAPHIC TRADE DIRECTORY WITH DATA COMPRESSION

FIELD OF THE INVENTION

The present invention relates generally to the field of computerized trade directory systems and more particularly to a system for a trade directory which includes advertising for individual subscribers and features text and image compression.

BACKGROUND OF THE INVENTION

Presently, manufacturers and competing trade directories are printed and comprise one or more volumes. The volumes tend to be thick and bulky.

Conventional trade directories are arranged by product category. Typically, the trade directory provides a company name and listing for each product. The company listing includes further information about the company. The company name and listing information is repeated for each product offered by a company. Since a company will typically have several products, the same information is repeated several times which leads to the thick and bulky nature of printed trade directories.

Competitors or manufacturers directories list companies by company name and also by product. In a conventional printed manufacturers' directory, the list by product category is typically eight times longer than the list by company name. Moreover, the list by product category repeats basically the same information. This substantially increases the size and publication cost of the directory.

While conventional trade directories provide an invaluable source of industrial information, their organization and structure lead to needless repetition of information. This increases the size and number of volumes required for a particular trade or manufacturing sector, which ultimately raises the publication cost for the printed directories. Furthermore, when a trade directory is updated, the existing trade directory volumes are essentially useless and therefore discarded. Since many directories are updated at least annually, this leads to waste and the necessity for recycling.

Another drawback of conventional trade directories arises from their very nature. Because trade directories are printed, they can only be searched manually, which can be inconvenient and time consuming. This problem is further compounded by the bulky and cumbersome nature of most directories.

BRIEF SUMMARY OF THE INVENTION

The invention therefore provides a system for computerizing a trade directory. A computerized trade directory according to the invention overcomes these disadvantages through the incorporation of a company listings directory stored on a series of floppy diskettes using compression methods according to the invention and a user interface which facilitates searching and retrieving information from the directory.

According to the invention, the information comprising the individual company listings is stored using a combination of tokenization and Huffman-encoding techniques. The compression technique is selected according to the purpose or content the information field in the company listing. It is a feature of the invention that any company listing can be decoded without having to decode the preceding company listings.

Another feature of the invention is the capability to include images, including photographs, in the company listing information.

The images are compressed and can comprise a mix of text, line drawings, photographs, spot colour and true colour portions. According to the invention, the images are compressed utilizing a combination of compression algorithms, with the particular compression algorithm being selected according to the type of image data in each portion of an image. Using the compression techniques according to the invention, the equivalent of sixty full screens of text and images, which may include not only spot colour but also monochrome and full-colour photographs, may be stored on a single high density 3.5 inch floppy diskette.

All of the data comprising the trade directory, including the images, can be used directly from the original diskettes or installed in compressed form on the hard drive of a computer. According to the invention, searched information is selectively unpacked as needed from the diskettes, or hard drive. This saves storage space eliminating the need for very high density mass storage devices, such as compact disk ("CD ROM") devices, and thereby allowing the directory according to the invention to be distributed to a wide range of users who simply need a computer having a conventional hard disk and a floppy disk drive.

Another feature of the invention is that images stored on diskette according to the invention are used to provide advertising for companies which have purchased the feature. The system automatically displays the advertising images when the associated company or product category is being searched. According to the invention, the advertisements can be classified according to relative importance which is used for selecting the advertisment to be automatically displayed by the system.

In a first aspect, the present invention provides a system for providing a trade directory on a computer having a display monitor, data entry means and disk storage, said system comprising: (a) database means for providing information associated with a plurality of listings

contained in the trade directory, said information being stored in compressed form on the disk and including company information and product information; (b) search means for searching said database means including means for searching said listings, means for searching selected company information, and means for searching selected product information; and (c) means for generating a screen for the display monitor, said means for generating including means for decoding information located by said search means and means for presenting said decoded information on said screen.

In a second aspect, the present invention provides a method for generating a compressed image from an image having text, line drawing, spot colour and continuous-tone portions, said method comprising: (a) providing a list of default colours for encoding the image; (b) identifying portions of the image corresponding to said list of default colours and encoding said portions using lossless compression; (c) identifying portions of the image not corresponding to said list of default colours and encoding said portions using lossy compression; (d) combining said lossless compressed portions and said lossy compressed portions to form a compressed mixed image file; and (e) storing said compressed mixed image file on a disk.

BRIEF DESCRIPTION OF THE DRAWINGS

For a better understanding of the present invention, and to show more clearly how it may be carried into effect, reference will now be made, by way of example, to the accompanying drawings which show preferred embodiments of the present invention, and in which:

Figure 1 is a schematic view of a trade directory system according to the present invention;

Figure 2 is a diagrammatic representation of a Main Screen for the system of Figure 1;

Figure 3 is a diagrammatic representation of a Search Screen which is generated by the system of Figure 1;

Figure 4 is a diagrammatic representation of a Product List screen which is generated by the system of Figure 1;

Figure 5 is a diagrammatic representation of a Company Listing screen which is generated in response to a product category search in the Index Screen of Figure 4;

Figure 6 is a diagrammatic representation of a Bookmark Screen according to the present invention;

Figure 7 is a diagrammatic representation of a Profile Screen which is produced by the system of Figure 1;

Figures 8(a) and 8(b) are diagrammatic representations of a full page advertisement which is stored and displayed by the system according to the present invention;

Figure 9 is a diagrammatic representation of a Help Screen which is generated by the system of Figure 1;

Figure 10 is a logical flow diagram showing the steps involved in displaying a company listing including images;

Figure 11 is a logical flow diagram for a routine which is called by the routine in Figure 10 to retrieve and decompress the text portion of the company listing;

Figure 12 is a logical flow diagram for a routine which is called by the routine in Figure 10 to display one of the images associated with a company listing;

Figure 13 is a logical flow diagram for a routine which is called by the routine of Figure 12 to locate the images associated with the company listing;

Figure 14 is a logical flow diagram for a routine which is called by the routine of Figure 13 to read and decompress the image file;

Figure 15 is a logical flow diagram for a routine which is called by the routine of Figure 14 to read a JPEG encoded image file according to the invention;

Figure 16 is a logical flow diagram for a routine which is called by the routine of Figure 14 to read a mixed-images file according

to the invention;

Figure 17 is a logical flow diagram showing the top level of an image compression algorithm according to the invention;

Figure 18 is a logical flow diagram showing a compression routine which is called by the compression algorithm of Figure 17 to compress an image;

Figure 19 is a logical flow diagram for a routine which is called by the routine of Figure 18 to partially run-length encode the image;

Figure 20 is a logical flow diagram for a routine which is called by the routine of Figure 19 to compress and store the "runs";

Figure 21 is a logical flow diagram for a routine which is called by the compression algorithm of Figure 17 to encode a block of pixels from the continuous-tone portions of the image;

Figure 22 is a logical flow diagram of the routines which are called to encode the continuous-tone portions of the image;

Figure 23 is a logical flow diagram of a routine for executing a Forward Discrete Cosine Transform;

Figure 24 is a logical flow diagram of a routine which compresses and stores the encoded continuous-tone portions of the image;

Figures 25(a) and 25(b) are logical flow diagrams for decoding a compressed mixed-image according to the invention;

Figure 26 is a logical flow diagram of a routine for an Inverse Discrete Cosine Transform; and

Figures 27 is a logical flow diagram showing the steps for decoding Huffman-encoded data.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

Reference is made to Figure 1 which shows in diagrammatic form a preferred embodiment of a system for a computerized trade directory 10 according to the present invention.

The system 10 according to the present invention comprises a computer 12 having a display monitor 14, a keyboard 16, and a disk storage device 18 for storing a trade directory 20. The computer 12 can be an IBM PC/AT® or compatible personal computer. Preferably, the computer 12 has a '386 or higher microprocessor architecture and includes at least 4 Mb of RAM. The computer 12 preferably also has a mouse 22. Preferably, the display monitor 14 is a VGA display which can display at least 32,768 colours.

The disk storage device 18 comprises a conventional mass storage device such as a hard disk or a floppy disk. For the system 10 shown in Figure 1, the storage device 18 is a hard disk which is installed in the computer 12. The computer 12 also includes a floppy disk drive 24 which accepts floppy diskettes.

The trade directory 20 comprises a main program module 26 and a series of volumes. The main program module 26 comprises a number of functions which control the operation of the system 10. The main module 26 provides a user interface to the system 10 and includes software for searching the volumes and decompressing data and images for presentation on the monitor 14. The volumes on the other hand contain company listings and images which are stored in compressed form to save disk space. The program module 26 is installed on the hard disk 18. The program module 26 is designed to run on a Microsoft Windows® platform as will be understood by those skilled in the art. As will be described in more detail below, the program module 26 provides a graphical user interface and searching tools for accessing and retrieving company and product information for the trade directory 20.

The system 10 also includes a program volume 28 which includes a number of files which are used by the program 26 to access and retrieve information from the volumes. The program volume 28 and program module 26 can be installed on the hard disk 18 or run from a floppy diskette. The first series of volumes, indicated by

reference 30, contain individual company database listings in compressed form. The second series of volumes 32 in the trade directory 20 contain the compressed images associated with selected company listings. The compressed company listing volumes 30 and image volumes 32 can be installed on the hard disk 18 or the accessed directly from floppy diskettes, e.g. one volume per diskette.

As will be described below, the present invention also comprises compression algorithms for producing the compressed company database listings and compressed images. In the Figure 1, the compression utilities are indicated generally by reference 34 and can be stored on the hard disk 18 or separately on a floppy diskette (not shown). In a typical application, the compression utilities 34 are used by a publisher to prepare the trade directory.

It is a feature of the invention that the company listings and images for trade directory 20 can be stored entirely on and run from a series of floppy diskettes, i.e. corresponding to volumes 1 to 9. Since a floppy disk drive 24 is a standard equipment for a personal computer 12, the trade directory 20 according to the invention has wide appeal which is further enhanced by the capability to store and retrieve images, including photographs, from a floppy diskette.

Referring to Figure 1, the program volume 28 in the trade directory 20 comprises the following files: a "Contents" file 36, a "Cities" file 38, a "Provinces" file 40, a "cities.dat" file 42, an "Index" file 44, an "index.dat" file 46, and a "Countries" file 48. The volume 28 can also include a "Bookmark" file 50 which is created by the program 26 as will be described below. The program volume 28 also includes a start-up screen file (not shown), which contains a full title screen and a secondary title screen. The system 10 displays the full title screen on initial start-up, and the secondary title screen is displayed in the upper-right corner of the main screen as will be described with reference to Figure 2.

The Contents file 36 contains information which is used by

the main module 26. The information includes the number of data/image volumes, e.g. volumes 1 to 9; the total number of listings in the trade directory 20; the first and last data record and image record in each volume 30 and 32; and the first and last company name in each volume 30 and 32.

The Cities file 38 contains a list of city names. The city names appear in alphabetical order of "country", "province/state" and "city". The Cities file 38 also includes a header containing the total number of names and the length of each name. The system 10 uses the Cities file 38 in conjunction with the Provinces file 40.

The Provinces file 40 contains a list of provinces/states and countries. For each province or state, the file 40 includes a cross-reference to the Cities file 38. The cross-reference comprises a pair of numbers which indicate the first and last city name for the particular province or state in the Cities file 38.

The cities.dat file 42 contains a list of record numbers, which identify the listings, i.e. company name and profile, associated with each city. The record numbers appear in the same order as the list of city names in the Cities file 38. As will be described in more detail below, the system 10 uses the list of record numbers as an index when searching by city/state/province or country.

The Index file 44 contains a list of product classification/sub-classification names in alphabetical order. Each product name appears in the format: "classification: sub-classification".

The index.dat file 46 contains another list of record numbers. The record numbers in the index.dat file 46 identify the listings, i.e. companies, associated with each product sub-classification. The record numbers appear in the same order as the list of product names in the Index file 44. As will be described in more detail below, the system 10 uses the index.dat file when searching by product classification or sub-classification.

The Countries file 48 contains a list of import/export country

names and names of imported products. The system 10 utilizes this file for tokenization purposes to reduce the space required in the individual data records to store the import/export information.

The Bookmark file 50 is generated by the bookmark feature of the system 10. As will be described in more detail below, the bookmark file 50 saves search criteria, including current position in the list, for use at a later time.

For the system 10 shown in Figure 1, there is the program volume or disk 28 and nine volumes of company listings and images, which can correspond to ten floppy diskettes. While the individual company listings and images can fill up to nine volumes, i.e. volumes #1 to #9, the actual number of volumes for company listings, number of volumes for images, and the subsequent total number of diskettes will depend upon the total amount of information contained in the directory.

As will now be described, the system 10 according to the invention provides the following features. First, the system 10 through the program 26 provides a graphical user interface. The graphical interface provides easy-to-use tools for searching and retrieving information contained in the directory 20. The program 26 also includes the facility to display information on how to use the directory 20, including a "Help" file and an "Index" of available products in the directory 20. Secondly, the program 26 includes search tools which provide a range of searching techniques which permit listings to be retrieved by company name, product classification, product subclassification, city, state, province or country. Thirdly, the program 26 also includes data and image decompression facilities for decoding and displaying information for the listings selected by the user. The data and image decompression facilities according to the invention can selectively unpack a compressed company listing or a compressed image without the need to convert the preceding records to decompressed form. This permits the data to be used in its original

form, either from the hard disk 18 or from a set of floppy disks.

The system 10 also includes text and image compression utilities 34 which allow a trade directory publisher, for example, to produce the compressed company listings volumes 30 and compressed image volumes 32.

The operation of the graphical user interface embodied in the system 10 will now be described with reference to Figures 2 to 9.

When initially run, the system 10 displays a title screen (not shown), which can be customized to the individual directory or to the publisher of the directory. The title screen is followed by a main screen 52 of the type shown in Figure 2. As shown in Figure 2, the main screen 52 comprises a companies listing scroll box 54, a selected company listing display box 56 and an image display box 58. The main screen 52 also includes a menu or command bar 60.

Referring to Figure 2, the command bar 60 includes the following commands: "Search" 62, "Index" 64, "Bookmark" 66, "Profile" 68, "Image" 70, "Help" 72, and "Exit" 74. The command bar 60 also includes an "About" 76 command. The main screen 52 also includes an icon command bar 78 which is located at the bottom of the companies listing scroll box 54. The icon command bar 78 has an icon command button corresponding to each of the commands in the command bar 60. There is a Search icon 80, an Index icon 82, a Bookmark icon 84, a Profile icon 86, First and Second Image icons 88,90, a Help icon 92, and an Exit icon 94. In response to clicking the First Image icon 88, the program 26 displays the first screen of the image, and clicking the Second Image icon 90 causes the program 26 to display the second screen of the image.

As shown in Figure 2, the companies listing box 54 includes a selection bar 96 for selecting a company listing for display. In operation, the user selects a company by "clicking" on the company name in the listing box 54. In response, the selection bar 96 appears across the selected company, and the program 26 retrieves a listing 98 and

image(s) 100 associated with the selected company. The program 26 displays the listing 98 in the listing display box 56 and the image 100 in the image display box 58. As will be described in more detail below, the program 26 automatically displays the images, which are selected according to the product specified during search and the size and importance of each advertisement in the selected portion of the database and a listing selected by the user.To select another company in the listing box 54, the user simply "clicks" on the name of the other company and the program 26 moves the selection bar 96 and displays the associated listing in the display box 56 and the image in display box 58.

Because a company listing can have more than one image, the company listing display box 56 includes a field 102 which indicates the number of images associated with the selected company. When a company is selected from the listing box 54, the program 26 automatically updates the field 102 with the number of images associated with the company listing.

The commands contained in the command bar 60 and command icon bar 78 comprise the user interface provided by the program 26, and as will now be described, the commands enable an user to conveniently find and retrieve information from the trade directory 20.

The Search command 62 (and icon 80) is used to invoke the "search" function provided by the system 10. In response to the search command 62 (or icon 80) being clicked, the program 26 prompts the user by displaying a Search Screen 200 (which is described with reference to Figure 3). In the Search Screen 200, the program 26 prompts the user for the company name, product category, and/or location of company. In response to the information entered by the user, the program 26 will generate a new list of company names which are displayed in the company listing box 54 on the main screen 52.

The Index command 64 (or icon 82) is used to display a list of

all product classifications and sub-classifications which are stored in an Index File. The list of product classifications and sub-classifications are displayed in a Product Index Screen 220 such as shown in Figure 4. The user can select one of the product categories (or sub-categories) to obtain a list of companies which produce the selected product.

The Bookmark command 66 (or icon 84) allows the user to create a "bookmark" for the currently selected company. The user can use the "bookmark" at a later time to return to the same listing. When a bookmark is created, the program 26 stores the search criteria and the position of the currently selected listing in the list of companies. In response to the Bookmark command 66, the program 26 prompts the user with a Bookmark Screen 240 as shown in Figure 6. As will be described below, the Bookmark Screen 240 allows the user to add, delete and select bookmarks.

The Profile command 68 (or icon 86) is used to display a Profile List Box or Screen 260 as shown in Figure 7. The Profile Screen 260 lists all of the text associated with the selected company listing.

The Image command 70 causes the program 26 to display the image(s) associated with the selected company to be displayed. In response to the image command 70, the program 26 displays a full page image as shown in Figures 8(a) and 8(b). According to the invention, a full page image can cover two screens. The program 26 displays a full page image as a top-half image 280a shown in Figure 8(a) and a bottom-half image 280b shown in Figure 8(b). Clicking the image icons 88 and 90, respectively, causes the program 26 to toggle between the two image portions. This feature of the present invention allows a full page image to be easily displayed on the standard sized display monitor 14 which is sold with most personal computers 12.

The Help command 72 (or icon 92) invokes a help function. In response to the user clicking the help command 72 or icon 92, the program 26 displays a Help Screen 300 as shown in Figure 9. The Help Screen 300, which can comprise additional screens (not shown),

includes information to "help" the user with the operation of the trade directory 20.

The Exit command 74 (or icon 94) allows the user to exit or leave the program 26. In a Windows® environment, exiting the program 26 returns the user to DOS. Lastly, the About 76 provides the user with information "about" the trade directory 20 and program 26, for example, the version number and release date.

Reference is again made to Figure 3 which shows the Search Screen 200 in detail. The Search Screen 200 is accessed by clicking the Search command 62 or search icon 80 on the main screen. As shown in Figure 3, the Search Screen 300 is displayed in modal style which will be understood by those skilled in the art. According to the invention, the trade directory 20 can be searched by product classification/sub-classification, company name and/or geographical location.

As shown in Figure 3, the Search Screen 300 has a Company Name edit box 302, a product Classification edit box 304, a City edit box 306, a State/Province edit box 308, and a Country edit box 310.

The Search Screen 300 also includes an Ok command button 312 and a Cancel command button 314. In response to the user clicking the Ok button 312, the program 26 executes a search of the trade directory 20 based on the information which has been entered in the edit boxes 302 to 310. To leave the Search Screen 300 without doing a search, the user simply clicks the Cancel button 314, and the program 26 returns to the main screen 52.

To perform a name search, the user positions the cursor in the name edit box 302 by clicking the mouse 22 (Figure 1). The user enters the company name by typing on the keyboard 16 (Figure 1). The user then clicks the Ok button 312, and the program 26 initiates a search based on the name or character string entered in the name edit box 302.

To search by classification or product, the user selects the classification edit box 304 and enters the name of a product (or product sub-category). In response to the user clicking the Ok button 312, the

program 26 begins searching for the company listings associated with the product name entered in the classification edit box 304.

A search by classification can also be done using the Index Screen 220 shown in Figure 4. The Index Screen 220 is displayed in modal style when the Index command 64 (or icon 82) is clicked. The Index Screen 220 lists all the product categories and sub-categories contained in the trade directory 20. As shown in Figure 4, the Index Screen 220 has a Category list box 222, a Subcategory list box 224, an Ok command button 226, a Cancel command button 228, and a Help command button 230. The Index Screen 220 also includes a row a alphabetic command buttons 232.

The program 26 will configure the Category list box 222 with a scroll bar 234 if the list of product names is larger than can be displayed in the box 222. Similarly, the program 26 will configure the Subcategory list box 224 with a scroll bar (not shown) if the number of subcategory names associated with the selected product classification exceeds the size of the Subcategory box 224. The classifications which initially appear in the box 224 are product classifications from "AA to AZ". (The program 26 divides the category index into sections, one for each letter of the alphabet.) To go to a different section of the category index, the user clicks the first letter of the category name in the alphabetic command buttons 232. The user can also change the category section by entering the first letter of the desired category name on the keyboard 16 (Figure 1).

As shown in Figure 4, the program 26 uses the Subcategory box 224 to display the subclassification product names associated with the selected product category. The program 26 uses a selection bar 236 to indicate the selected product category, e.g. "Absorbents". To select another product category, the user can click on the product name after entering the first letter using the alphabetic command buttons 232. Similarly, the user selects the desired subcategory by clicking one of the names listed in the box 224. To initiate a search for companies

associated with the selected product category and/or subcategory, the user clicks the Ok button 226. To exit the Index Screen 220 without doing a search, the user simply clicks the Cancel button 228.

The index by product subclassification is stored in the trade directory 20 in the two files: Index 44 and Index.dat 46 (Figure 1). The Index file 44 comprises one continuous alphabetical list of product classifications and subclassifications. The main classification appears first, followed by the individual subclassifications in classification: subclassification format as shown in the Subcategory box 224 in Figure 4. The program 26 divides the continuous list contained in the Index file 44 into 26 alphabetical sections and into separate product category and subcategory. The index.dat file 46 contains a list of record numbers indicating which company listings are associated with each of the products contained in the Index file 44.

The program 26 displays the results of a search in the Main Screen 52 as shown in Figure 5. (As will be described below, the display of a company listing can be controlled by the display of advertising images.) As shown in Figure 5, the program 26 displays the search results as follows. The names of companies which carry the product are displayed in the list box 54. In this example, there is one company which is associated with the product category: "Absorbents", and the company listing is indicated by a selection bar 97. The information associated with the selected company is displayed in the box 56 and indicated by reference 99. The advertising image 101 for the selected company is displayed in the box 58. The advertising image 101 could also be displayed in full screen format as shown in Figures 8(a) and 8(b).

To control the display of advertising images 100, the program 26 uses the following fields which are included in each listing in the trade directory 20 (i.e. volumes 1 to 9): BOLD LISTINGS, AD, IMAGE, IMAGE2, and AUTO-DISPLAY. The BOLD LISTINGS field specifies the number of product categories which are to be displayed in bold type. The "AD" field specifies the number of product categories for which

images are to be displayed. The "IMAGE" and "IMAGE2" fields specify the original file name of each image. The "AUTO-DISPLAY" field specifies the relative size/importance of an advertisement image over a range of display options.

It is another feature of the invention that the relative size/importance of an advertisement can be specified from a range of zero to seven, where larger/more important images have a higher number. At the high end, the program 26 automatically displays the advertising image in full screen format (see Figures 8(a) and 8(b)) in response to a search. At the other end of scale, the advertising image is displayed in the box 58 on the main screen 52 (see Figure 5). While a listing may contain up to twenty product names, the images stored for the listing may consist of advertising which promotes only one or a selected few of these items. Therefore, before displaying an advertisement, the program 26 checks if the image relates to the product(s) which the user is searching.

Once the desired information has been found, the user can create a "bookmark" to the selected information. The user selects the bookmark command 66 (or clicks the bookmark icon 84) and in response the program 26 displays the Bookmark Screen 240. The program 26 displays the Bookmark Screen 240 in modal form.

As shown in Figure 6, the Bookmark Screen 240 includes an edit box 242, a bookmark list box 244, an "Add" command button 246, a "Delete" command button 248, a "Select" command button 250, and a "Cancel" command button 252. The bookmarks which appear in the list box 244 are stored and retrieved from the Bookmark file 50 (Figure 1).

To add a bookmark, the user positions the cursor in the edit box 242 (e.g. by clicking the mouse 22) and the name of the bookmark, e.g. "Absorbents", is entered using the keyboard 16. Once the name has been entered, the user clicks the Add button 246 and the program 26 will store the bookmark in the bookmark file 50. The program 26 also

updates the list box 244 to show the newly created bookmark, e.g. "Absorbents".

To select a bookmark, the user clicks the mouse 22 on the name of the desired bookmark, e.g. "New York, USA", displayed in the list box 244. The user then clicks the Select command button 250. In response, the program 26 recalls the information associated with the selected bookmark from the bookmark file 50. The program 26 uses this information to retrieve and display the corresponding company listing information from the trade directory 20.

Referring still to Figure 6, to remove a bookmark, the user clicks the mouse 22 on the name of the bookmark, e.g. "Conventions-New York", displayed in the list box 244. The user then clicks the Delete command button 248, and in response, the program 26 deletes this bookmark from the bookmark file 50.

According to the invention, a company listing can contain additional information which cannot all be displayed in the display box 56 and the image display box 58. This additional information is accessed through the profile feature. The Profile Screen 260 shown in Figure 7 is accessed from the Main Screen 52 by selecting the profile command 68 or clicking the profile icon 86.

As shown in Figure 7, the Profile Screen 260 provides additional information about the selected company, e.g. "Sorbent Products Co. Inc.". The additional information comprises, for example, a product summary indicated by reference 262 and standard industrial classification numbers indicated by reference 264. The information shown in the Profile Screen 260 can also include information on imports, exports, personnel, corporate structure, and company finances.

Reference is again made to Figure 9 which shows the Help Screen 300 provided by the system 10. The Help Screen(s) 300 provide "help" to a user and are accessed by selecting the Help command 72 or clicking the Help icon 92. As shown in Figure 9, the Help Screen 300 comprises a display box 302 and a menu or command bar 304. The

display box 302 provides information about the trade directory 20 and instructions on how to use the system. Certain words in the display box 302 appear underlined or in a different colour (not shown). Clicking the mouse on these words will cause the program 26 to display additional information. The command bar 304 includes a "Contents" command 306, a "Search" command 308, a "Back" command 310, a "History" command 312, and "Right" and "Left" scroll buttons 314,316. These commands allow the user to easily navigate through multiple Help Screens.

Reference is again made to Figure 2. In operation, when switching from one listing to another, the program 26 will do one of the following operations. First, if searching by product category, the images associated with the newly selected company listing apply to this category and the auto-display field is not zero, then the program displays the images in full screen format for five seconds each. Second, if the company listing includes images and the user has not selected a category which contradicts the products in the images, the program 26 will display the first image in the upper-right corner of the main screen 52 (Figure 2). Third, if the listing has no images, or a product category is being searched which does not match any of the products indicated for the images, the program 26 will display a default image in the upper-right hand corner of the main screen 52. The logical steps executed by the program 26 to display a company listing (i.e. data record) are shown in flow chart form in Figures 10 and 11.

Reference is made to Figure 10 which shows a routine for displaying a record 320. The display record routine 320 generates the contents of the main screen 52 (Figure 2) including the company listing information 98 displayed in box 56 and the image 100 displayed in box 58. The display routine 320 also generates the other contents of the main screen 52 such as the icon bar 78 and icons 80 - 94, and title bars. Accordingly, the display routine 320 is called by the program 26 when the main screen 52 needs to be redrawn, including when a different

company listing 98 is selected or searched by the user, and including when the image is displayed in full screen format 280a,280b (Figure 8).

When called by the program 26, the display routine 320 first checks if a new record, i.e. company listing, is to be displayed in block 322. If yes, the display routine 320 removes the previous listing from the main screen 52 and retrieves the data record associated with the new listing from the trade directory 20 (block 324). Since the data records are stored in compressed form, the display routine 320 calls another routine 325 (Figure 11) which decompresses the data record and generates the company listing information 98 (Figure 2). The operation of the routine 325 is described below under the heading "Data Compression" and with reference to Figure 11.

Next in block 326, the display routine 320 checks if the new company listing includes any images. If there are new images, then according to the invention, the images are displayed automatically in a full-screen format. This feature is used to present advertising information associated with a company listing. According to the invention, two images can be displayed in a full-screen format, and the display for the images can be defined using a variable "drawimage" to display the first image, the second image, or both first and second images. In block 328, the display routine 320 determines if the first image (i.e. image 280a in Figure 8) is to be displayed and if yes, the first image is displayed in full-screen format in block 330. To display an image, the display routine 320 calls a routine 331 as shown in Figure 12. According to the invention, the program 26 displays a full-screen image 280a or 280b for five seconds, and then the main screen 52 is displayed. In block 332, the display routine 320 sets the five second "time-out" and if both images are to be displayed, the routine 320 also loads the second image (e.g. image 280b in Figure 8). The second image is then automatically displayed for five seconds after the first image has timed-out.

In block 334, the display routine 320 determines if the second

image is to be displayed (i.e. instead of the first image or after the first image). If the second image is to be displayed, the display routine 320 displays the second image 280b (as shown in Figure 8(b)) and sets the five-second timer as indicated in blocks 336 and 338 of Figure 10.

On the other hand, if there are no images associated with the new company listing, or the display routine 320 is being called to redraw the main screen 52, then in block 340, the display routine 320 determines how the existing image should be displayed. If the existing image is to be displayed in full-screen format, the routine 320 proceeds to blocks 328 and 334 as described above. If the image 100 is to be displayed in the main screen 52 (as shown in Figure 2), the display routine 320 proceeds to blocks 342 and 344 in Figure 10. In block 342, the routine 320 generates the list box 54, command bar 60 and the icon bar 78 (and icons) for the main screen 52, and in block 346, the display routine 320 adds the title boxes for the screen 52. In block 346, the display routine 320 checks if the list box 54 is empty. If the list box 54 is not empty, the display routine 320 displays the company listing information 98 in box 56 for the selected company in the list 54 (block 348 of Figure 10). In block 350, the image (e.g. image 100) associated with the selected company listing is displayed in the display box 58 as shown in Figure 2.

According to the invention, an image 100 which is displayed in the corner of the main screen 52 (Figure 2) may also be displayed as a full screen in response to a user command, i.e. by selecting the image command 70 or clicking one of the image icons 88,90. This will cause the program 26 to re-display the image 100 for one minute on the full screen or until a key in the keyboard 16 or the mouse 22 is pressed.

Data Compression

Another feature of the system 10 according to the present invention is the data compression of information for each company listing. According to the invention, individual company listings in

volumes 1 to 9 (Figure 1) are stored using a combination of Huffman encoding and tokenization. Huffman encoding techniques for data compression are known and within the understanding of one skilled in the art.

The compression techniques according to the invention are selected based on the purpose or usual content of each field in a data record for a company listing. It is a feature of the present invention that any record may be decoded at any time, without having to decode or decompress the preceding records, which in turn, allows the trade directory 20 to be run directly from diskettes.

As described with reference to Figure 1, the compressed company database listings are stored in the first few data volumes 30 and the compressed images are stored in separate volumes 32 which appear later in the directory 20. The contents file 36 provides the program 26 with a list or index of the contents of each volume 1 through 9.

Individual records in the database volumes are compressed by selectively applying one of three possible lists of tokenization values or one of two different Huffman codes. One Huffman code is selected for compressing plain-ASCII text, and one Huffman code is selected for plain-ASCII numbers.

The product names and export product names are tokenized using a list of product classifications and subclassifications which are defined in the index file 44. This allows the system 10 to replace each line by a two-byte token. The three place name fields i.e. city, province/state, country, are tokenized together using the city file 38 and the province file 40 as a list of places. The import/export country names and imported product names are tokenized using the country file 48.

The system 10 compresses facsimile numbers by storing a count of digits which match the main telephone number, followed by the actual digits of the rest of the number, where the numbers are

encoded using the Huffman code. Numeric values which are used to control operation of the program 26, for example the number of products listed in bold type and the number of products associated with the display advertisement, are stored as pure binary numbers and therefore must not contain non-numeric characters in the original data. Typically any other fields are stored as Huffman-encoded text. The system 10 selects the appropriate Huffman table according to the expected contents of the field, which may be either mostly numbers or mostly text. It will be appreciated that this will not prevent text from appearing in numeric fields or vice versa, it only assures that numeric fields store numbers efficiently while text fields store letters efficiently.

According to the invention, instead of storing the same company listing once under each product category, as is done in conventional printed trade directories, all information for the same manufacturing facility is contained in one data record. The system 10 provides two indices, the cities files 38,42 and the index files 44,46. The index.dat file 46 lists all records associated with each product classification and subclassification in the index file 44, and the cities.dat file 42 lists all records for manufacturers in each city listed in the cities file 38. The system 10 does not use an index by company name because the main database used by the program 26 is already sorted by name.

The cities 38 and index 44 files also allow the program 26 to use tokenization of place, i.e. city, province/state, country, and category, and also tokenization of product and export names. Instead of storing a line of text to indicate the name of a product, or three for city, state/province and country, the program 26 uses a field (e.g. two bytes) which stores the line of the Index file 44 (Figure 1) containing the name of the desired product or the line in the Cities file 38 containing the name of the desired place.

The index by product subclassification is stored as two files: Index 44 and Index.dat 46. The format used by these files 44,46 is similar to that of cities 38 and cities.dat 42, except that there is no list of

provinces/states and countries.

The Index file 44 is one continuous alphabetical list of product classifications and subclassifications. The main classification appears first, followed by the individual subclassifications in the classification according to the format "classification: subclassification". The index.dat file 46 contains a list of record numbers indicating which company listings are associated with each of these products.

According to the invention, where a company listing indicates a specific product subclassification, the index.dat file 46 will list only the subclassification, i.e. the associated main classification is not repeated. Whenever a search is done for manufacturers of one of the main classifications of products, the program 26 will also search manufacturers of each individual product subclassification in this main class. As described above, the system 10 divides the index into 26 alphabetical sections, i.e. A-Z, and into separate product classification and subclassification lists. To the system 10, the data files appear as one continuous list.

The system 10 uses the Country file 48 to store the names of importing/exporting countries and imported products. The system 10 uses this file for tokenization and there is no associated index.

Reference is made to Figure 11 which shows the "datarecord" routine 325 for reading and decoding a compressed company listing from one of the volumes 30 (Figure 1). In block 352, the routine 325 calls another routine "getrecord" which reads the compressed data for the data record, i.e. company listing. The getrecord routine (not shown) first searches the Contents file 36 to determine which volume 30, i.e. #1 to #9, contains the data record for the selected company. The getrecord routine then determines the position of the desired data record in the volume 30 and reads the compressed data into memory. The routine 325 then proceeds to decompress the data record.

In block 354, the routine 325 decodes the product classifications, the company name, and advertising fields. The product

classifications are decoded using the list of product classifications and subclassifications which are defined in the Index file 44. The company name is decoded according to the Huffman code which was used to compress the name. The advertising fields contain information such as number of products associated with the advertising images for the company and are read as pure binary numbers.

Referring still to Figure 11, in block 356, the routine 325 checks if the data record should be displayed as a partial record. The partial record format is used by the program 26 to display the company name in the list box 54 for a particular product category, for example. To save processing time for a partial record, the routine 325 only decodes the company name, product classifications and advertising fields because the company listing information is only needed for display in box 56 if the company is selected from the list box 54. If the data record is not to be displayed as a partial record, the routine 325 proceeds to decode the company address, the phone number, and the company profile information in block 358. As described above, the company address is compressed by tokenization using the Cities 38 and cities.dat 42 files. The phone number and facsimile number are decoded using the Huffman code and tokenization for the facsimile number as described above. The company profile information typically comprises compressed text and this is also decoded according to the tokenization and Huffman code which was used for encoding the text. For the system 10 according to the present invention, the program 26 includes Huffman-tree tables ALPHA and NUM for decoding the compressed information comprising the company listings. The program 26 also includes a Huffman-tree table for decoding data for continuous-tone images and a Huffman-tree table for decoding data for mixed images. A routine for decoding Huffman encoded data is illustrated in Figure 27. The complete implementation and use of Huffman-encoding techniques is based on the nature of the data to be compressed and within the understanding of those skilled in the art of

data compression.

According to the invention, each company listing can include a designation indicating the type of company. Each of the company types is tokenized using a single letter as follows: "contractor" tokenized as "C"; "distributor" tokenized as "D"; "engineers" tokenized as "E"; "hotel" tokenized as "H"; "manufacturer" tokenized as "M"; and "service" tokenized as "S". In block 360, the routine 325 converts the company type field to the full word.

Once the routine 325 has finished retrieving and decompressing the data record, control is returned to the display record routine 320 (Figure 10). As described above, the display record routine 325 generates the main screen 52 and displays the decoded information for the data record in the appropriate boxes of the main screen 52. For example, partial records are displayed in the list box 54, the company listing information is displayed in the display box 56, and the advertising image 100 is displayed in the box 98.

Image Compression

It is a feature of the present invention that company listings can include images. The images can contain display advertising, text, line drawings, logos and photographs all on the same screen. The system 10 according to the invention features an image compression process which provides the capability to store such images including full-colour pictures and images having spot colour.

The image compression process according to the invention determines which portions of an image contain continuous-toned photographs, and applies lossy compression on these portions while using loss-less compression on the remainder of the image. This produces images with compression ratios equal to or better than those obtained using purely lossy compression while eliminating compression losses outside the segments of the image used for photos. In addition, the time required to decode the resulting image is reduced.

It will be appreciated that this is a significant feature since the system 10 can be operated directly from diskette.

The first step in the compression involves identifying which portions of the image are to be loss-lessly encoded. To identify lossless-encodable data, the image compression utility uses a pattern based on a very narrow range of pixel values which appear in the image. While colour photographs contain thousands of different pixel values, text and line drawings will typically contain only a few, e.g. white, black, one or two shades of grey and one or two spot colours. The compression utility therefore identifies the different image segments by using a list of pixel values which occur most often in the image. Most of the pixels in the photographs will not match any colour on this list, but all of the pixels for the other image components will match. The remaining continuous-tone image pixels can be identified as being a smaller number of pixels surrounded by unknown colours on each side. The list of colours could be generated by reading the entire image and keeping a list containing a running count of the number of pixels of each colour. Once the list becomes full, the least-used colour can be discarded. Once this process is complete, the most used fifteen colours could be selected and used to determine which portions of the image are not continuous-tone photographs. While such an approach may be suitable, in the preferred embodiment it is not used because the colours for text, background and spot colour change little from one image to the next. Almost all of the images contain advertising copy in black on white text, titles in black or spot red, 50% or 75% grey used as additional spot colours, and the remainder of the image data composed of continuous-tone photos.

To save processing time for the compression utility, it is preferred to use a list of default spot colours. For the few cases where the text foreground or background colours are non-standard, the compression utility includes the capability for manual selection of desired colours.

In operation, the image compression algorithm comprises the following steps for encoding a complete image. (The steps for encoding an image according to the invention are described in more detail below with reference to the Figures.) First, a list of spot colours is made from the default colours listed in the compression utility and any extra colours specific to this image. Second, an attempt is made to run-length encode the entire image, with a value "OTHER" being used to replace any colour not on the list of spot colours. For each block, e.g. 8 x 8 pixels or 16 x 8 pixels if under-sampling is used for continuous mode, the compression utility sets a variable "HAS_OTHER" to true if the block has any value with "OTHER". Almost-adjacent runs of OTHER separated by a few pixels of a spot colour are combined into one longer run of OTHER. This run-length encoded image is then Huffman encoded and written to disk. Next, if any OTHER values were found by the utility, each block containing OTHER is encoded using known techniques for JPEG lossy (baseline DCT) compression. However, according to the invention, the standard JPEG headers are not used and the Huffman tables are constant as will be understood by those skilled in the art.

A feature of the compression algorithm according to the invention is the capability of providing 45:1 compression using 24-bit decompressed bitmaps as the input, containing a mixture of advertising copy and photographs. (Portions of ISO draft standard 10918-1 pertaining to the sequential baseline DCT JPEG image encoding process are within the understanding of those skilled in the art and are incorporated herein by this reference.)

The compression algorithm will now be described with reference to the logical flow diagrams shown in the Figures. As described above, the compression algorithm for producing images according to the invention comprises encoding the image using a combination of lossless compression techniques for text, line drawing and spot colour portions of the image and lossy compression

techniques for continuous-tone portions of the image. The compressed data is then stored in an output file which is decompressed by the program 26 (see above). Using the compression algorithm according to the invention, a compression ratio of 45:1 can be achieved using 24-bit uncompressed bitmap data for images comprising advertising copy and photographs.

The top level of the compression algorithm is shown in Figure 17 and indicated generally by reference 500. The compression algorithm 500 opens an input bitmap file (step 502) and an output file (step 504). The input file contains uncompressed 24-bit bitmap data corresponding to a scanned, i.e. digitized, image. Each 24-bit data sample corresponds to a pixel in the image. The output file will store the compressed image file created by the algorithm 500. In step 506, the compression algorithm 500 calls a compression routine 508 as shown in Figure 18.

Reference is next made to Figure 18 which shows the compression routine 508. The compression routine 508 converts the uncompressed 24-bit bitmap data corresponding to the mixed-image portions, i.e. text, line drawings and spot colour, into a compressed mixed image file which is stored in the output file. The compression routine 508 creates the compressed mixed-image file by trying to run-length encode the entire 24-bit bitmap. Run-length encoding involves identifying "runs" of bitmap data (i.e. pixels) having the same colour from the list of spot colours and encoding these runs by storing the length of the run (i.e. number of pixels) and colour in the spot colour list. The encoding of the run by storing the length and colour index effectively compresses the pixels as will be understood by those skilled in the art. If the run of bitmap data comprises a colour not on the spot colour list, then it is stored as an "OTHER" type run. According to the invention, OTHER type runs correspond to continuous-tone portions and are encoded (i.e. compressed) using known lossy JPEG compression techniques.

As shown in Figure 18, the compression routine 508 first determines the maximum number of continuous-tone blocks in the bitmap data in block 510. In block 512, the compression routine adds the RGB (Red-Green-Blue) values for any extra spot colours added by the user. Next in block 514, the compression routine 508 calls a lossless compression routine 516 which is shown in Figure 19.

Referring to Figure 19, the lossless compression routine 516 generates a run-length encoded portion for the mixed image. (The continuous-tone portions of image are added later by the compression routine 508 in Figure 18.) The lossless compression routine 516 first calls a routine ("addheader") which adds a header to the mixed-image file as indicated by block 518. The header defines the file as a mixed-image file, and indicates the image width and height in pixels, provides a list of spot colours in the image, and also indicates the width and height of continuous-tone image blocks. Then for each pixel (i.e. 24-bit bitmap data), the lossless compression routine 516 compares the colour of the pixel to that of the previous pixel (block 522). As indicated in block 518, if there is a change in the colour between pixels, then the lossless compression routine 516 calls another routine 526, "addrun", (shown in Figure 20) which adds the run, i.e. same colour pixels, to the output file.

Reference is made to Figure 20 which shows the "addrun" routine 526. When a run of pixels is generated by the lossless compression routine 516, the addrun routine 526 stores the run in arrays "runs[]" and "indices[]" indicated in block 528. If a short run of colour (i.e. corresponding to text, line drawings and spot colour) separates two longer runs of OTHER bitmap data (i.e. corresponding to continuous-tone portions of the image), then the routine 526 combines the two OTHER runs and the colour run into one long OTHER run, as indicated in blocks 530 to 538. This results in a continuous-tone portion of an image being encoded entirely as a continuous-tone image even though a few pixels match one of the spot colours on the list. Once the

number of allowed runs have been stored, the routine 526 writes the oldest run to the output file on disk (blocks 540 and 542). If the entire image has been run-length encoded, then the routine writes all the remaining runs to the output file on disk (blocks 544 and 546). The addrun routine 526 then returns to the lossless compression routine 516 and the run-length encoding process is repeated (block 548) until the end of the bitmap, i.e. last pixel, is reached (block 550).

Referring back to Figure 18, once the compression routine 508 has completed the run-length encoding, i.e. lossless compression of the bitmap data, the routine 508 checks for continuous-tone portions of the image, i.e. OTHER bitmap data, as indicated in blocks 552 and 554. If there are no continuous-tone image portions, control is returned to the main loop of the compression algorithm 500 (Figure 17).

According to the invention, if there are any continuous-tone portions in the image, these are encoded or compressed using lossy techniques. As shown in block 556, the routine 508 adds a second header to the output file which will contain data on the continuous-tone portions of the image, including colour type (i.e. true-colour with 256-colour palette, monochrome or true colour), number of entries in colour palette, definition of colours to be used in the 256-colour palette, quantization tables, block size and undersampling information, as will be understood by those skilled in the art. Next the routine 508 encodes the continuous-tone image portions using JPEG compression as follows. In block 558, the routine 508 compresses a block of pixels corresponding to continuous-tone image data by calling a "writeblocks" routine 560 as shown in Figure 21. The continuous-tone image data comprises a series of 8 x 8 pixel blocks (or 16 x 8 pixel block, if undersampling is being utilized) and the routine 560 starts with the first block of pixels as indicated at step 562. At step 564, the routine 560 encodes each 8 x 8 pixel block by first calling a "getpixels" routine. The getpixels routine reads one 8 x 8 pixel block and uses the defined colour values to convert each primary colour (R,G,B) in the pixel data to

luminance, red difference or blue difference values (depending which field is being read and in known manner, each pixel includes a luminance and chrominance field). The values produced by getpixels are stored in memory and the routine next calls a "writeblock" routine 566 as shown in Figure 22.

Referring to Figure 22, the writeblock routine 566 calls three other routines FDCT 568, quantize 570, and writezigzag 572.

The FDCT routine 568 which is shown in Figure 23 performs the forward discrete cosine function. In the context of the invention, the forward discrete cosine function transforms the pixel data into a format which occupies less memory. The transformed pixel data is further compressed by applying Huffman-encoding. (To display a compressed image, the inverse process is applied to the encoded data as will be described below.) The pixel array comprising the image is broken into blocks or squares of 8 x 8 pixels, and the FDCT is applied to transform each block into transformed values zz[0..63], where zz[0] represents the average value of the 64 pixels and zz[63] represent progressively higher frequency changes in the 8 x 8 block of pixels. The forward discrete cosine transform is defined as follows:

$$zz_{v,u} = \frac{1}{4} C_u C_v \sum_{x=0}^{7} \sum_{y=0}^{7} s_{y,x} \cos\frac{(2x+1)u\pi}{16} \cos\frac{(2y+1)v\pi}{16}$$

where:

$C_u, C_v = 1/\sqrt{2}$ for u,v = 0 and are 1 otherwise

To save time, "$s_{y,x}$" is pre-initialized to $2048 * C_u \cos((2x + 1)u\pi/16)$ and used as a "look-up" table as will be understood by those skilled in the art. In addition, an alternate form of the forward discrete cosine transform can be utilized to allow the calculation to be done with three "nested loops" instead of four. As shown in Figure 23, the routine 568 performs the actual calculations in two stages using integer arithmetic. In block 574, each $temp_{i,j}$ is calculated as follows:

$$temp_{i,j} = -\Sigma (image_{i,k} * cs_{j,k}/256) \text{ for } k = 0 \text{ to } 7$$

And in block 576, the transform is calculated as follows using the values of $temp_{i,j}$ determined above.

$$zz_{i,j} = \Sigma (temp_{k,j} * cs_{i,k}/65536 \text{ for } k = 0 \text{ to } 7$$

The result, i.e. zz[], produced by the FDCT routine 568 is processed by a quantize routine. The quantize routine rounds off each element in the zz[] array in order to reduce the number of bits required for storage in the compressed image file. The quantize routine divides each element, i.e. zz[i], using a quantization table, i.e. qtable[i].

Reference is next made to Figure 24 which shows the writezigzag routine 572. The routine 572 converts the values in the array zz[] into a Huffman-encoded form in which the number of bits used to represent each value is Huffman-encoded, and the actual bits are written directly to the output file. According to the routine 572, first the size and magnitude of the element zz[0] are written (block 578). The routine 572 checks the remaining elements, i.e. zz[1... 63], for runs of zeros (block 580). A run of zeroes at the end of the array zz[] is not stored as indicated in blocks 582 and 584. As indicated in blocks 586 and 588, the remaining elements in the array zz[] are written in compressed form as a Huffman-encoded byte is used to represent the value of an element, where the high nibble is the number of zeroes which precede the element, and the low nibble is the number of bits used to represent the value of element and the individual bits used to represent the magnitude of the value of the element in the array zz[]. The routine 572 ends once all 64 values of array zz[] have been compressed (block 584), or once all non-zero values of array zz[] and a terminating code, e.g. 00 byte, have been written (block 590).

To display a mixed image compressed according to the invention, the program 26 essentially reverses the above procedure as will now be described with reference to Figures 12 to 16.

Reference is made to Figure 12 which shows the steps comprising the "show image routine" 331. The routine 331 is called by the program 26 in the "displayrecord" routine 320 (Figure 10). The showimage routine 331 uses two other routines: "openimagefile" 331 (Figure 13) and "readimage" 335 (Figure 14) to read an image 100 from disk (e.g. volume #9) and convert it to an uncompressed 24-bit bitmap, which can be displayed on the monitor 14.

As shown in Figure 12, the showimage routine 331 first checks if the previous image is to be displayed (block 364). If the previous image is being displayed, then the routine 331 moves directly to block 366. If the image is new, then before it can be displayed, the image must be retrieved from the disk (i.e. images volume(s) 32). In block 368, the routine 331 discards the previous image and reads and uncompresses the new image. To open and read the image, the routine 331 calls the "openimage" routine 331 and "readimage" routine 335 (as will be described with reference to Figures 13 and 14). After the execution of block 368, the image will be uncompressed and stored in a device-independent form. In block 370, the routine 331 determines the number of colours available to display the image on the monitor 14. For example, if fewer than 64 colours are supported, the routine 331 sets the "palette" to monochrome, or if 256 colours are available, the routine 331 sets the palette to 256 colours. Next in block 372, the routine 331 converts the bitmap data comprising the image into device-dependent form for use with the graphics interface installed in the computer 12.

Referring still to Figure 12, in block 374 the routine 331 determines whether the image is to be displayed as a full screen. It will be remembered that the display size for the image can be set according to the "AUTO-DISPLAY" field described above. If the image is full

screen, the routine 331 calculates screen position for the image in block 376 and then clears the screen in block 378. If the image is not a full screen display, the routine 331 determines the position for the image in the display box 58 (Figure 2) as indicated in block 380. In block 382, the routine 331 converts the image into the colours available in the palette as will be within the understanding of one skilled in the art. The routine 331 then copies the image to the monitor 14 according to the position determined in block 380 (or block 376 for a full screen image).

Reference is made to Figure 13 which shows the "openimage" routine 333 in more detail. The routine 331 calls openimage 333 to locate the desired image in the images volume(s) 32 (Figure 1). In block 386, the openimage routine 333 first checks if the image to be displayed is the default image. The default image is stored in a known bitmap file and the routine 333 does not need to search the image volume(s) 32. The routine 333 opens the bitmap file and returns to the showimage routine 331 which will then display the default image.

Referring still to Figure 12, if the image is not the default image, then the routine 333 will search the image volume(s) 32. Each image is assigned and stored according to a record number. Since there can be more than one image volume 32, each volume 32 includes an index of the images, i.e. record numbers, contained in the volume. To read an image, the routine 333 first finds and opens the volume 32, i.e. disk, containing the image (block 390). The routine 333 then skips over the preceding images to find the location of the image and compressed image data stored in the volume (block 392). The compressed image record is read into memory and the routine 333 returns control to the showimage routine 331.

To read the compressed image record, the showimage routine 331 calls the "readimage" routine 335 shown in Figure 14. The function of the "readimage" routine 335 is to convert the compressed image record into a device-independent bitmap form. To do this, the routine

335 must first determine the type of image: Windows bitmap, JPEG image, or mixed-image (i.e. combination of continous-tone images and mixed text and line drawings). The image type is determined by reading the first two bytes in the image file (block 394). If the image is stored as a straight bitmap file (block 396), then it is not necessary to decompress the image and the data can be read directly into memory (block 398). If the image is compressed in JPEG format (block 400), then in block 402 the routine 335 calls a "readjpg" routine 337 shown in Figure 15 to convert the JPEG format image to an uncompressed bitmap form, i.e. 24-bit. If the stored image comprises a "mixed-image" (i.e. containing text and continuous-tone portions) as determined in block 404, then the routine 335 calls a "readmi" routine 339 shown in Figure 16 to convert the mixed-image to an uncompressed and device-independent bitmap file (block 406).

Reference is next made to Figure 15 which shows the "readjpg" routine 337 in more detail. The function of the readjpg routine 337 is to convert an image which has been compressed in JPEG format into a device-independent Windows® bitmap file. The bitmap file is then displayed according to the graphics capabilities supported by the computer 12 and display 14.

As shown in Figure 15, the readjpg routine 337 reads the first two bytes of the compressed image file in block 408. The routine 337 then decodes the two bytes according to a decision-tree comprising decision blocks 410 to 416.

If the two bytes contain ff,db (hex), then the routine 337 calls a "quantizetable" routine in block 411. The quantize routine decodes the quantization table section in a JPEG file header as will be understood by one skilled in the art. This section of the JPEG file header is read as indicated in blocks 411 and 418. The quantize table routine is only used when decoding JPEG type images. It is not used for decoding mixed images.

If the two bytes contain ff,c0 (hex), then the routine 337 calls a "startframe" routine in block 413. The startframe routine reads the "start-of-frame" section in the JPEG header file as will be within the understanding of one skilled in the art.

Referring still to Figure 15, if the two bytes contain ff,c4 (hex), the routine 337 calls a "Huffman" table read routine in block 415. The Huffman read routine reads the Huffman tables from the JPEG header file. The Huffman tables are used to decode the Huffman encoded data as will be understood by one familiar with the well-known Huffman encoding techniques.

If the two bytes contain ff,da (hex), then the routine 337 calls a "startscan" routine in block 417. The startscan routine reads the start-of-scan section in the JPEG file header. The scan portion of the JPEG file contains the Huffman encoded image data. In block 419, the routine 337 reads and decodes the Huffman encoded image data and copies the decoded image data into a bitmap stored in memory. If an error occurs in decoding the image (block 420), an error message is generated in block 421. Otherwise, the routine 337 returns to the calling readimage routine 335 in Figure 14. The bitmap file containing the decoded image is then available for display by the program 26.

Reference is next made to Figure 16 which shows the "readmi" routine 339. The "readmi" routine 339 is called if the image is a "mixed-image" type file, i.e. an image comprising continuous-tone portions and text, line drawing or spot colour portions. The function of the readmi routine 339 is to decompress the mixed-image which was encoded using the compression method described above. In block 423, the routine 339 reads the image size and list of possible spot colour information from the header portion of the image file. Next in block 424, the routine 339 determines the width and height of the continuous-tone portion in the image. In block 425, the routine 339 generates a blank bitmap file in memory according to the image size determined in

block 423. The routine 339 will store the decompressed image data in the bitmap file.

As shown in Figure 16, the readmi routine 339 first runs through the entire image (block 426) decoding the portions of the image encoded using lossless compression, i.e. black-and-white text, line drawings and/or spot colour. Any continuous-tone portions are marked as "other" and decompressed after the lossless decoding. As described above, the text, line drawing and spot colour portions of the image were first run-length encoded and then further compressed using Huffman-encoding. Thus, each byte of lossless encoded image data is first decompressed according to the Huffman-encoding (block 427). Next in block 428, the routine 339 determines if there is a colour change in the "run". If there is a colour change, then the routine 339 changes the colour and reads the next byte of image data as indicated in block 429. (If the colour indicates an OTHER colour, i.e. corresponding to a continuous-tone portion, the routine 339 marks the run as OTHER and decodes it later.) If the run is longer than the length of a line, the routine 339 splits the "run" into a number of smaller runs (block 430). The length of a line in the image is defined in the header of the image file.

Referring still to Figure 16, in block 431, the routine 339 calls another routine to add the run, i.e. string of same colour pixels, to the bitmap file. The "addpixelrun" routine adds one run of colour to the bitmap file. The number of pixels for the colour written to the bitmap file are determined by the run length (block 432). In block 433, the routine 339 checks whether the colour of the run is white. If the run colour is white, the routine 339 sets the default colour for the next run to BLACK in block 434. Otherwise, the routine 339 sets the default colour for the next run to WHITE in block 435. The routine 339 repeats the steps in blocks 427 to 435 until the end of the screen, i.e. image, is reached.

Once the end of screen is reached, the routine 339 decodes the continuous-tone portions of the image as shown in blocks 436 through 439. The blocks for the continuous-tone portions were marked as OTHER by the routine 339 as described above. The routine 339 repeats the decoding process until all OTHER blocks are done (block 436). To decode a continuous-tone block, the routine 339 first calls a routine in block 437 once which reads the header for the continuous-tone block. In block 438, another routine is called repeatedly to read the continuous-tone block and convert it to bitmap form.

The header information for a continuous-tone block includes the following: number of colours (i.e. monochrome, full colour or full colour with 256 colour palette included); quantization table(s); number of 8 x 8 pixel blocks for continuous-tone portion to be read horizontally; and number of 8 x 8 pixel blocks for continuous-tone portion to be read vertically. For mixed-image files, the encoded image data corresponding to the continuous-tone portion is not further compressed once the JPEG baseline sequential encoding process is done.

Once the information from the header has been read, the routine 339 calls the routines which will decode (i.e. decompress) the encoded continuous-tone image. In block 438, the routine 339 first calls a readblocks routine 341 as shown in Figure 25(a). The readblocks routine 341 calls another routine in block 439 which decodes the continuous-tone data. In block 440, the routine 341 adds the decoded continuous-tone data (returned by routine in block 439) to the bitmap in memory which is used to display the image as described above. It will be remembered that the continuous-tone portion of a mixed image is encoded as a series of image blocks each having at least 8 x 8 pixels, (or more if undersampling has been used).

Reference is next made to Figure 25(b) which shows a routine 343 for decoding the continuous-tone data in the image block. The routine 343 as shown in Figure 25(b) decodes the continuous-tone data by reversing the operations performed to compress the continuous-

tone data as described above. The routine 343 first reads the compressed continuous-tone image block into an input buffer and start processing at the upper-left corner of the image block (as indicated in blocks 441 to 443). Next in block 444, the routine 343 calls another routine to decode the binary and Huffman-encoded data. In blocks 445 and 446, the routine 343 determines if the image is to be displayed in colour. The routine 343 then proceeds to decompress the image data by undoing the division done during lossy image compression and performing an Inverse Discrete Cosine Transform in order to convert the compressed image data, i.e. luminance and colour difference information, into uncompressed RGB pixel or bitmap data (blocks 447 and 448).

A routine 345 for performing the Inverse Discrete Cosine Transform is shown in Figure 27. As will be understood by one skilled in the art, the routine 345 performs an inverse discrete cosine transform (IDCT) on the "dequantized" image data in block 447 and 448 (Figure 26(b)) to produce an 8 x 8 block of pixel values for one field in the continuous-tone image. The routine 345 also converts the intensity and red and blue colour difference information (i.e. compressed image data) to RGB values. If undersampling was used when the image data was compressed, the routine 345 stretches the image block, i.e. horizontally and/or vertically, as indicated in Figure 26.

Referring to Figure 26, the routine 345 first initializes buffers: temp[] and sum[] to zero in block 449. In block 450, the routine 345 next converts the dequantized image data into 8 x 8 format using xzigzag[] and yzigzag[] to identify where each pixel element would appear in the 8 x 8 matrix. According to the Inverse Discrete Cosine Transform, the value of each pixel (x,y) is defined as follows:

$$IDCT(x,y) = \frac{1}{4} \sum_{u=0}^{7} \sum_{v=0}^{7} C_u C_v zz_{v,u} \cos\frac{(2x+1)u\pi}{16} \cos\frac{(2y+1)v\pi}{16}$$

where:

$C_u, C_v = 1/\sqrt{2}$ for u,v = 0 and are 1 otherwise; and

zz contains the dequantized values decoded from the compressed image file

To save time, the IDCT is performed using pre-calculated values for a table "cstable(x,u)" = $C_u \cos((2x+1)u\pi/16)*4096$. These values have been multiplied by 2^{12} and stored as integers to reduce the time required to perform the inverse Discrete Cosine Transform calculations.

As shown in Figure 26, the inverse DCT is done in two steps. The first step is shown in block 451 and produces an intermediate result which is stored in temp[u][y]. The second step is shown in block 452 and uses temp[u][y] to produce the inverse transformed data which is stored in the array sum[][]. Once the IDCT calculations have been completed, the routine 345 shifts all entries in sum[][] 3 bits to the right and copied into an array result[][] to provide the original 8 x 8 block of pixels (block 453). If the block was undersampled, the block must be stretched to the original number of pixels. In blocks 454 and 455, the block is stretched horizontally and in blocks 456 and 457, the block is stretched vertically. Next in blocks 458 and 459, the routine 345 converts each pixel into its R,G,B components using known techniques as will be understood by one skilled in the art. Once all the pixels in the block have been processed, the routine 345 returns. The process is repeated until the entire compressed image has been decoded, i.e. each pixel is converted into its R,G,B components.

Once the data and images associated with a company listing have been decompressed, the program 26 can proceed with constructing and displaying the screen 52 (Figure 2) as described above. If the advertising image is to be displayed as a full screen, then the program 26 will display the image in full screen format as shown in Figures 8(a) and 8(b).

It will be evident to those skilled in the art that other embodiments of the invention fall within its spirit and scope as defined by the following claims.

WE CLAIM:

1. A system for providing a trade directory on a computer having a display monitor, data entry means and disk storage, said system comprising:

(a) database means for providing information associated with a plurality of listings contained in the trade directory, said information being stored in compressed form on the disk and including company information and product information;

(b) search means for searching said database means including means for searching said listings, means for searching selected company information, and means for searching selected product information; and

(c) means for generating a screen for the display monitor, said means for generating including means for selectively decoding information located by said search means and means for presenting said decoded information on said screen.

2. The system as claimed in claim 1, wherein at least some of said listings include an image having a display mode identifier, and said search means includes means for retrieving an image associated with a selected listing, and said means for generating includes means responsive to said display mode identifier for presenting said image according to said display mode identifier.

3. The system as claimed in claim 2, wherein said image is stored in compressed form and can include mixed text and continuous-tone portions, said mixed text portions being compressed using a lossless compression format and said continuous-tone portions being compressed using a lossy compression format.

4. The system as claimed in claim 3, wherein said image further includes line drawing and spot colour portions, said line drawing and spot colour portions being compressed using a lossless compression format.

5. The system as claimed in claim 2, wherein said image comprises continuous-tone portions and is compressed utilizing JPEG compression format.

6. The system as claimed in claim 1, wherein said search means includes means for inserting a company name and searching said database means for said listing associated with said inserted company name.

7. The system as claimed in claim 6, wherein said search means includes means for inserting a product classification and searching said database means for said listings associated with said inserted product classification.

8. The system as claimed in claim 7, wherein said search means includes means for inserting a geographical location and searching said database means for listings located in said geographical location.

9. The system as claimed in claim 8, wherein said geographical location includes a country identifier.

10. The system as claimed in claim 8, wherein said geographical location includes a state identifier.

11. The system as claimed in claim 9 or 10, wherein said geographical location includes a city identifier.

12. The system as claimed in claim 7, wherein said search means includes means for providing a list of product classifications and means for displaying said list of product classifications on said screen.

13. The system as claimed in claim 12, wherein said list of product classifications includes a list of product sub-classifications for selected product classifications.

14. The system as claimed in claim 1, wherein said means for generating includes means for providing on said screen a list of listings located by said search means.

15. The system as claimed in claim 14, wherein said means for generating includes means for providing said company information in a window on said screen.

16. The system as claimed in claim 2, wherein said means for generating includes means for displaying said image in an image window on said screen.

17. The system as claimed in claim 2, wherein said means for generating includes means for displaying said image as a full screen when said display mode identifier defines a full-screen mode.

18. The system as claimed in claim 17, wherein said means for generating includes timeout means for displaying said image as a full screen for a pre-determined period of time and said generating means returning to said first screen after the expiry of said pre-determined period of time.

19. The system as claimed in claim 18, wherein said means for generating is responsive to an input from the data entry means for

aborting display of said image as a full screen and displaying said first screen.

20. The system as claimed in claim 1, wherein said search means further includes means for marking selected information located by said search means, and said search means having means responsive to said marking information for retrieving said selected information at a later time.

21. The system as claimed in claim 20, wherein said means for marking includes means for inserting a product classification to mark selected information located by said search means.

22. The system as claimed in claim 21, wherein said means for marking includes means for inserting a company name to mark selected information located by said search means.

23. The system as claimed in claim 22, wherein said means for marking includes means for inserting a geographical location to mark selected information located by said search means.

24. The system as claimed in claim 23, wherein said means for marking includes means for combining any of said product classification, company name and geographical location.

25. The system as claimed in claim 20, wherein said means for marking includes means for inserting a user defined label to mark selected information located by said search means.

26. The system as claimed in claim 1, wherein said means for generating includes means for providing a profile screen for displaying

additional company information, said means for providing a profile screen being responsive to an input from the data entry means.

27. The system as claimed in claim 14, wherein said means for generating includes means for selecting a listing from said list and displaying one or more images associated with said selected listing.

28. A method for generating a compressed image from an image having text, line drawing, spot colour and continuous-tone portions, said method comprising:

(a) providing a list of default colours for encoding the image;

(b) identifying portions of the image corresponding to said list of default colours and encoding said portions using lossless compression;

(c) identifying portions of the image not corresponding to said list of default colours and encoding said portions using lossy compression; and

(d) combining said lossless compressed portions and said lossy compressed portions to form a compressed mixed image file; and

(e) storing said compressed mixed image file on a disk.

29. The system as claimed in claim 28, further including the step of Huffman-encoding portions of the compressed mixed image file before storing on a disk.

30. The system as claimed in claim 29 wherein said lossy compression comprises JPEG encoding.

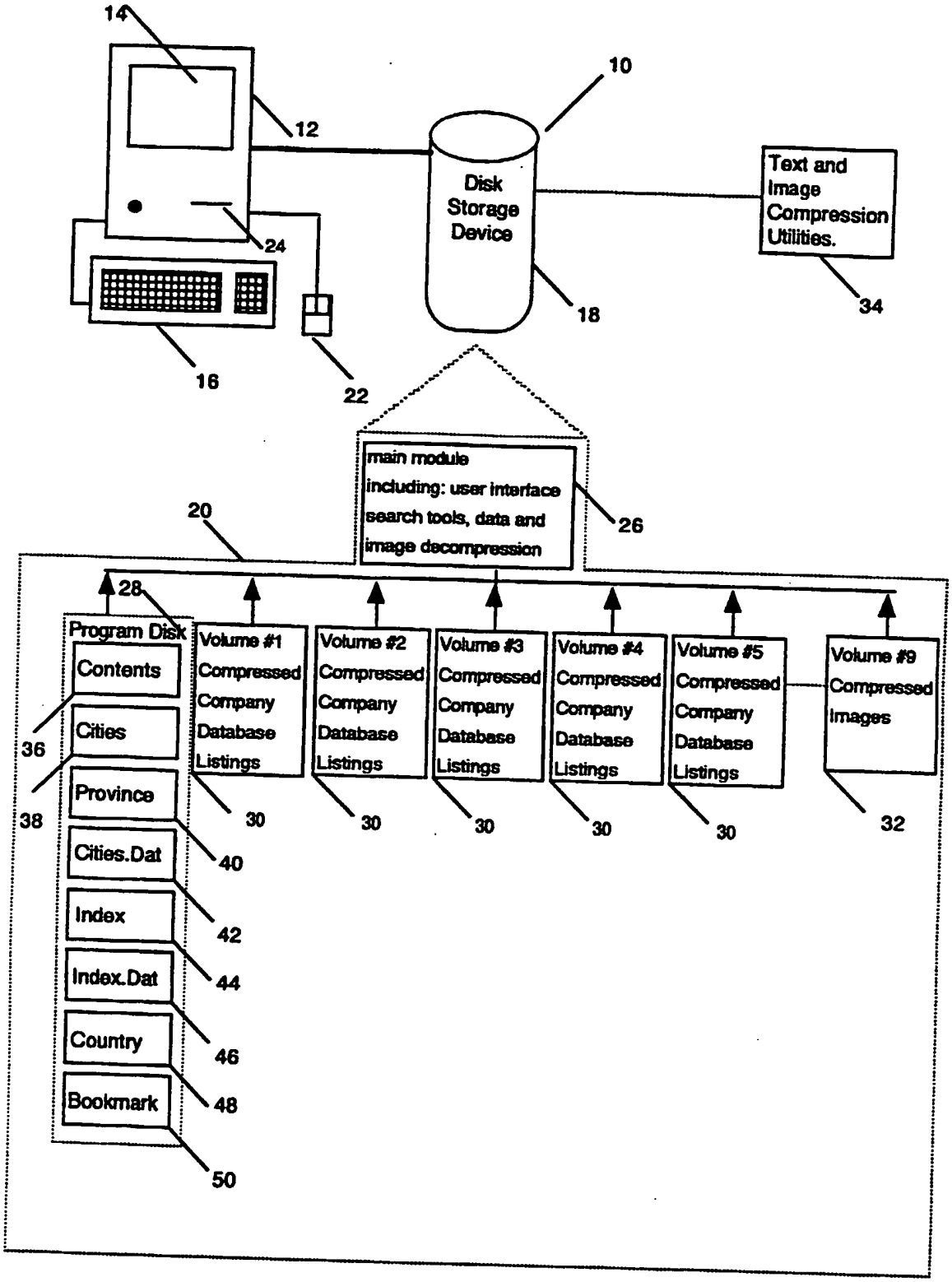


FIGURE 1

SUBSTITUTE SHEET

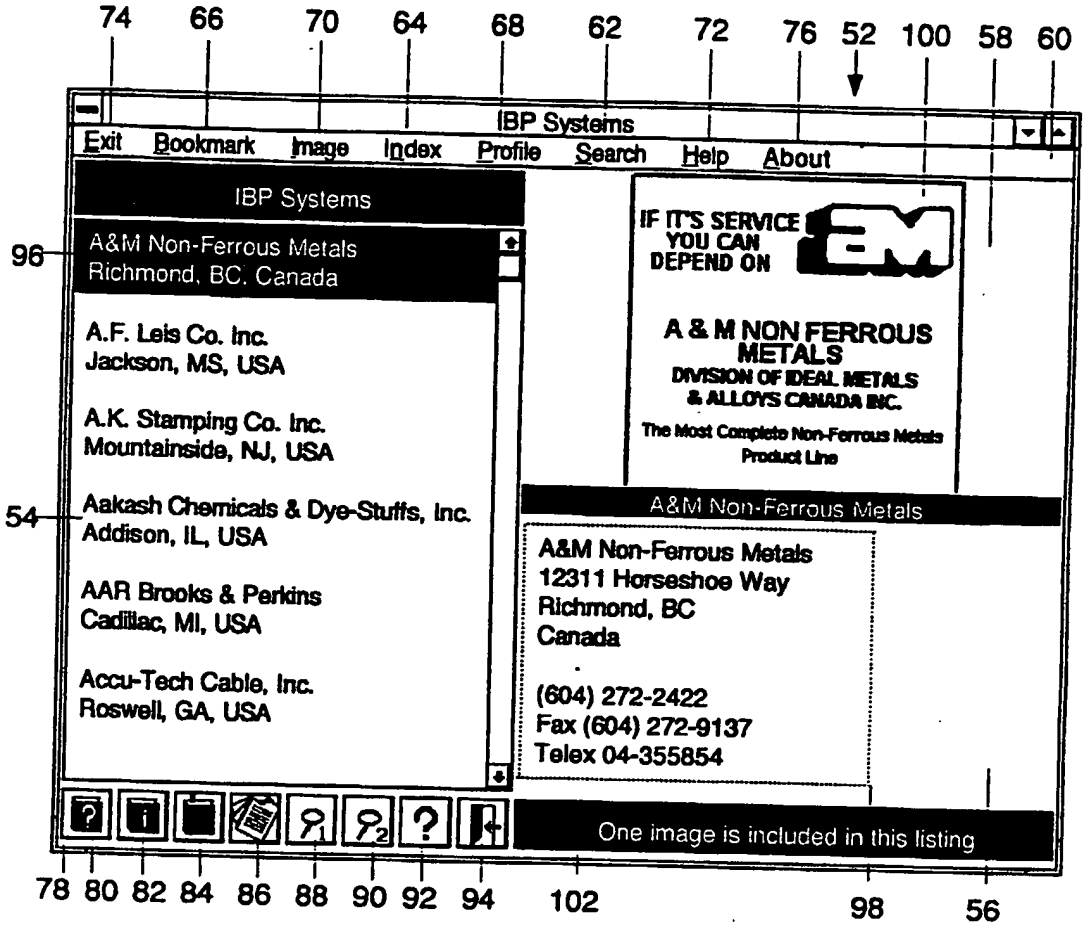

74
66
70
64
68
62
72
76
52
100
58
60
IBP Systems
Exit
Bookmark
Image
Index
Profile
Search
Help
About
IBP Systems
96
A&M Non-Ferrous Metals
Richmond, BC. Canada
A.F. Leis Co. Inc.
Jackson, MS, USA
A.K. Stamping Co. Inc.
Mountainside, NJ, USA
54
Aakash Chemicals & Dye-Stuffs, Inc.
Addison, IL, USA
AAR Brooks & Perkins
Cadillac, MI, USA
Accu-Tech Cable, Inc.
Roswell, GA, USA
IF IT'S SERVICE YOU CAN DEPEND ON
A & M NON FERROUS METALS
DIVISION OF IDEAL METALS & ALLOYS CANADA INC.
The Most Complete Non-Ferrous Metals Product Line
A&M Non-Ferrous Metals
A&M Non-Ferrous Metals
12311 Horseshoe Way
Richmond, BC
Canada
(604) 272-2422
Fax (604) 272-9137
Telex 04-355854
One image is included in this listing
78
80
82
84
86
88
90
92
94
102
98
56

FIGURE 2

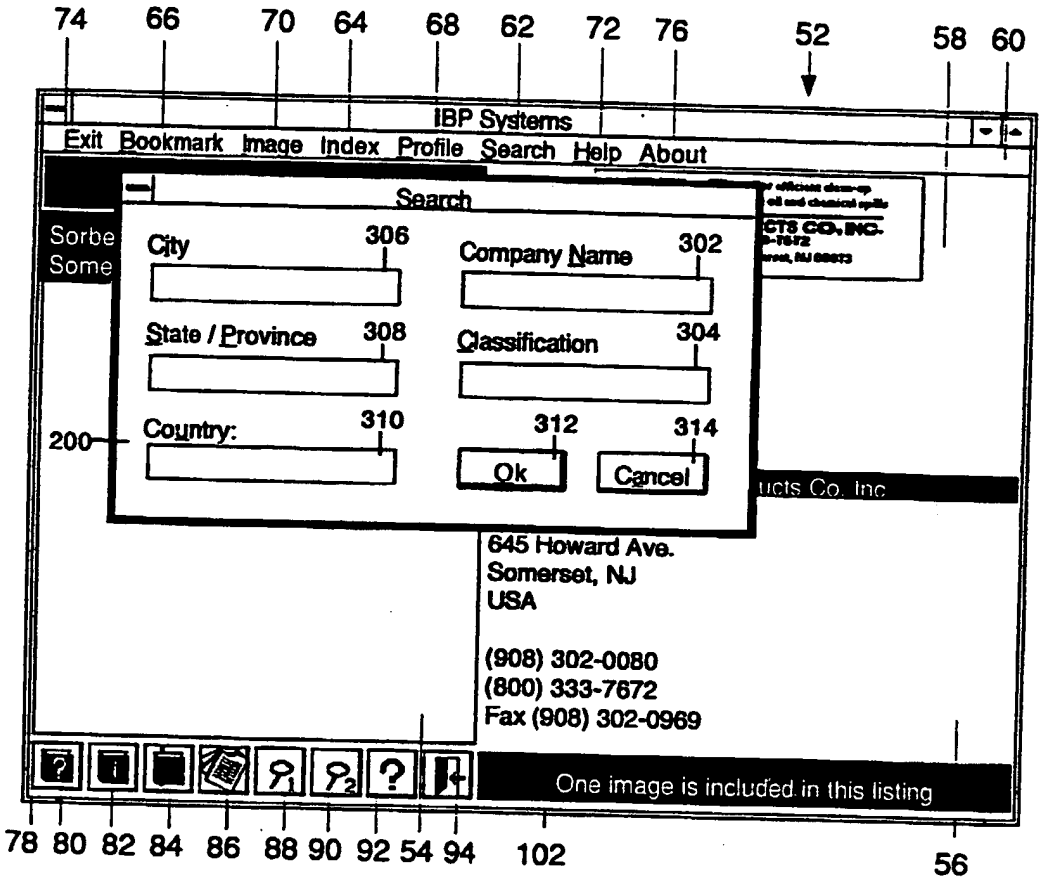

74
66
70
64
68
62
72
76
52
58
60
IBP Systems
Exit Bookmark Image Index Profile Search Help About
Search
Sorbe
Some
City
306
Company Name
302
State / Province
308
Classification
304
200
Country:
310
312
314
Ok
Cancel
ucts Co. Inc
645 Howard Ave.
Somerset, NJ
USA
(908) 302-0080
(800) 333-7672
Fax (908) 302-0969
One image is included in this listing
78 80 82 84 86 88 90 92 54 94 102
56

FIGURE 3

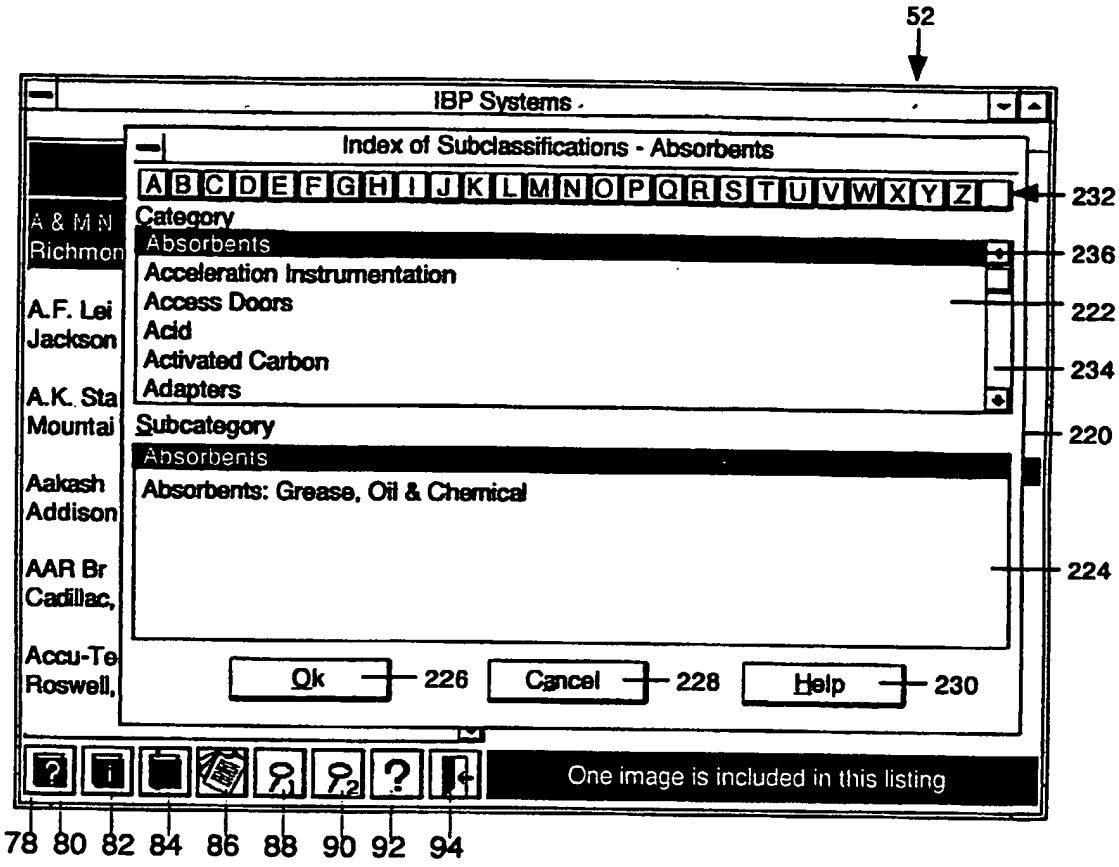

52
IBP Systems
Index of Subclassifications - Absorbents
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
232
Category
Absorbents
236
Acceleration Instrumentation
Access Doors
222
Acid
Activated Carbon
234
Adapters
Subcategory
220
Absorbents
Absorbents: Grease, Oil & Chemical
224
Ok
226
Cancel
228
Help
230
A & M N
Richmon
A.F. Lei
Jackson
A.K. Sta
Mountai
Aakash
Addison
AAR Br
Cadillac,
Accu-Te
Roswell,
One image is included in this listing
78 80 82 84 86 88 90 92 94

FIGURE 4

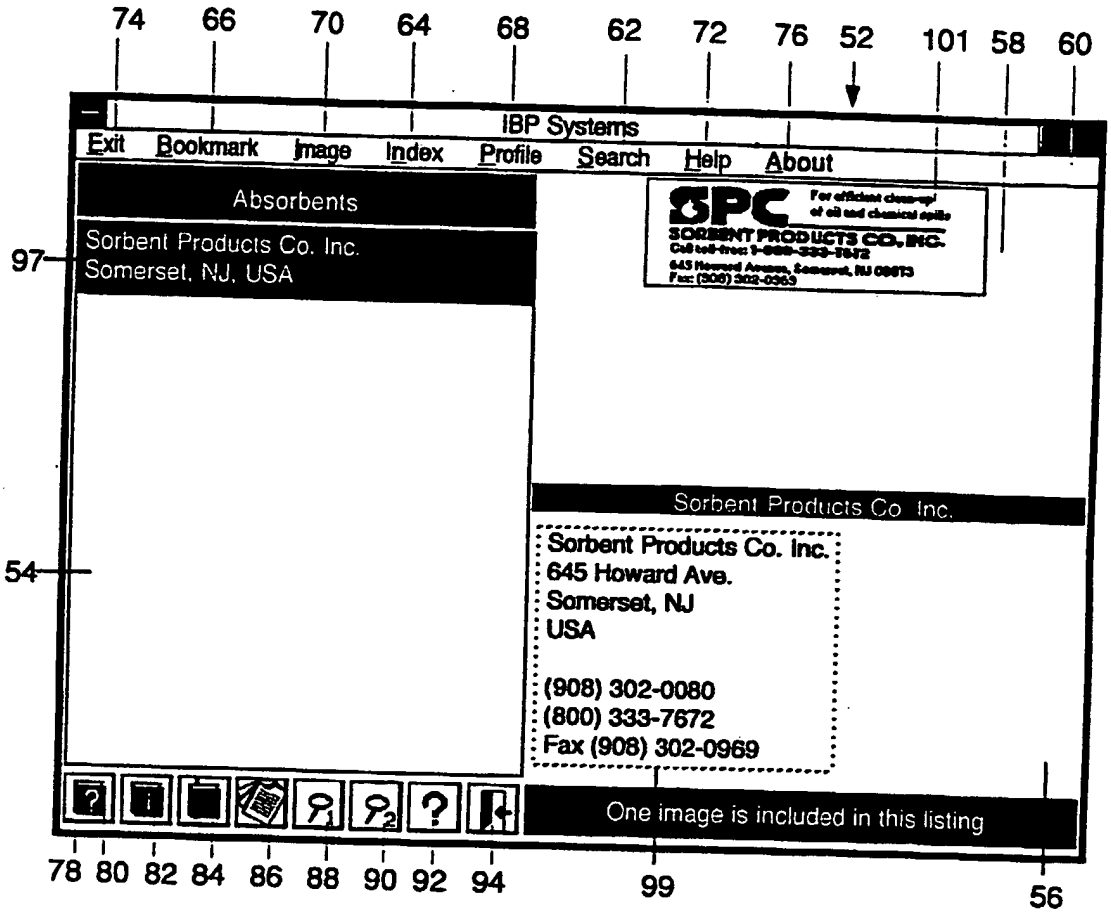

74
66
70
64
68
62
72
76
52
101
58
60
IBP Systems
Exit
Bookmark
Image
Index
Profile
Search
Help
About
Absorbents
97
Sorbent Products Co. Inc.
Somerset, NJ, USA
SPC
SORBENT PRODUCTS CO., INC.
54
Sorbent Products Co. Inc.
Sorbent Products Co. Inc.
645 Howard Ave.
Somerset, NJ
USA
(908) 302-0080
(800) 333-7672
Fax (908) 302-0969
One image is included in this listing
78
80
82
84
86
88
90
92
94
99
56

FIGURE 5

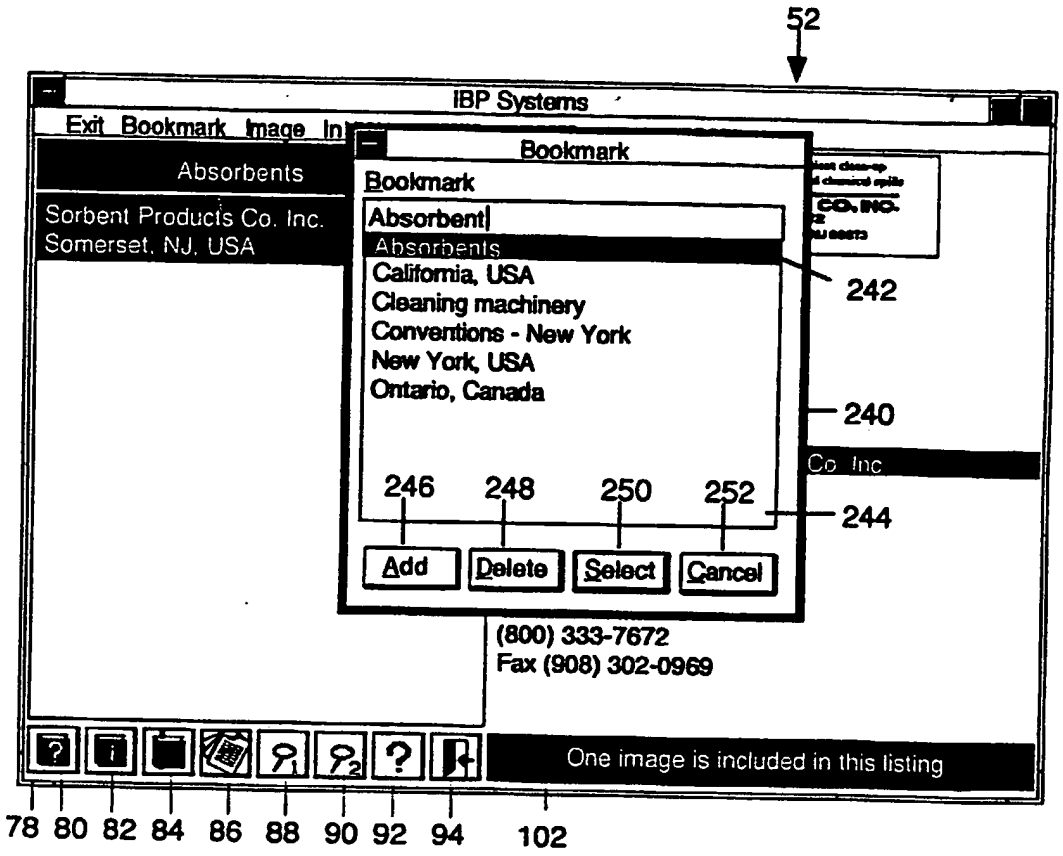

52
IBP Systems
Exit Bookmark Image
Absorbents
Sorbent Products Co. Inc.
Somerset, NJ, USA
Bookmark
Bookmark
Absorbent
Absorbents
California, USA
Cleaning machinery
Conventions - New York
New York, USA
Ontario, Canada
242
240
Co Inc
246
248
250
252
244
Add
Delete
Select
Cancel
(800) 333-7672
Fax (908) 302-0969
One image is included in this listing
78 80 82 84 86 88 90 92 94 102

FIGURE 6

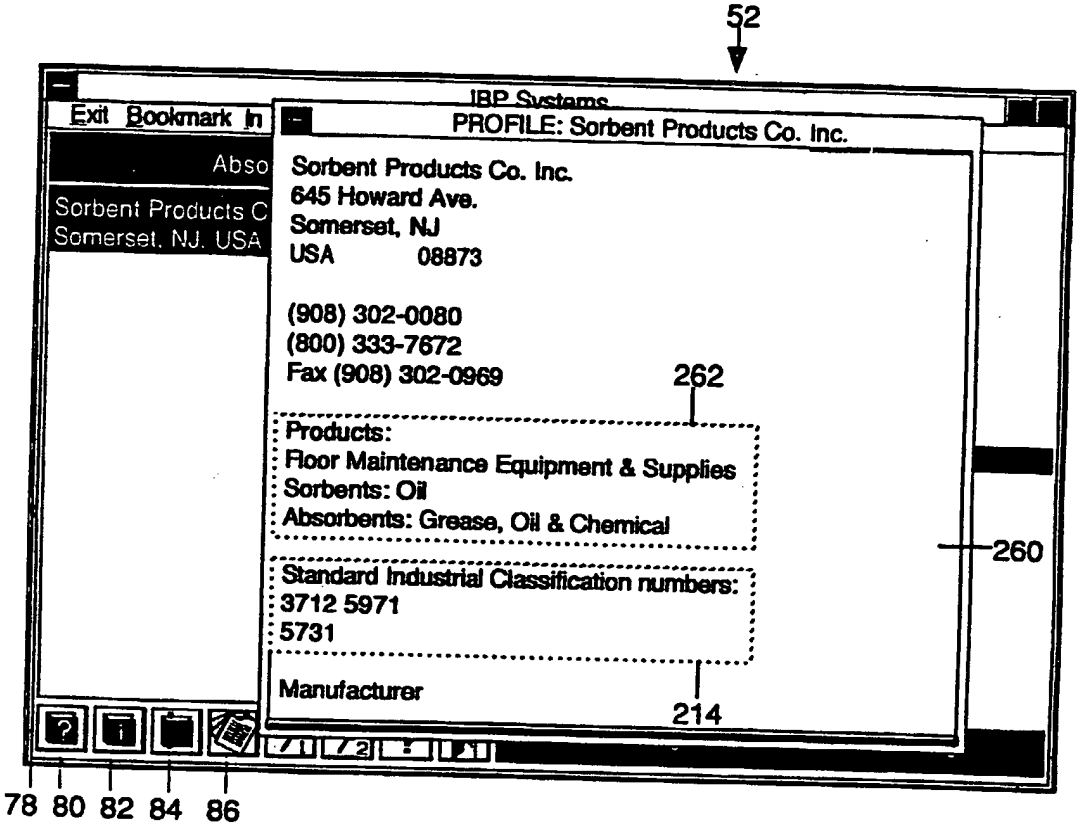

52
IBP Systems
Exit Bookmark In
Abso
Sorbent Products C
Somerset, NJ, USA
PROFILE: Sorbent Products Co. Inc.
Sorbent Products Co. Inc.
645 Howard Ave.
Somerset, NJ
USA 08873
(908) 302-0080
(800) 333-7672
Fax (908) 302-0969
262
Products:
Floor Maintenance Equipment & Supplies
Sorbents: Oil
Absorbents: Grease, Oil & Chemical
260
Standard Industrial Classification numbers:
3712 5971
5731
Manufacturer
214
78 80 82 84 86

FIGURE 7

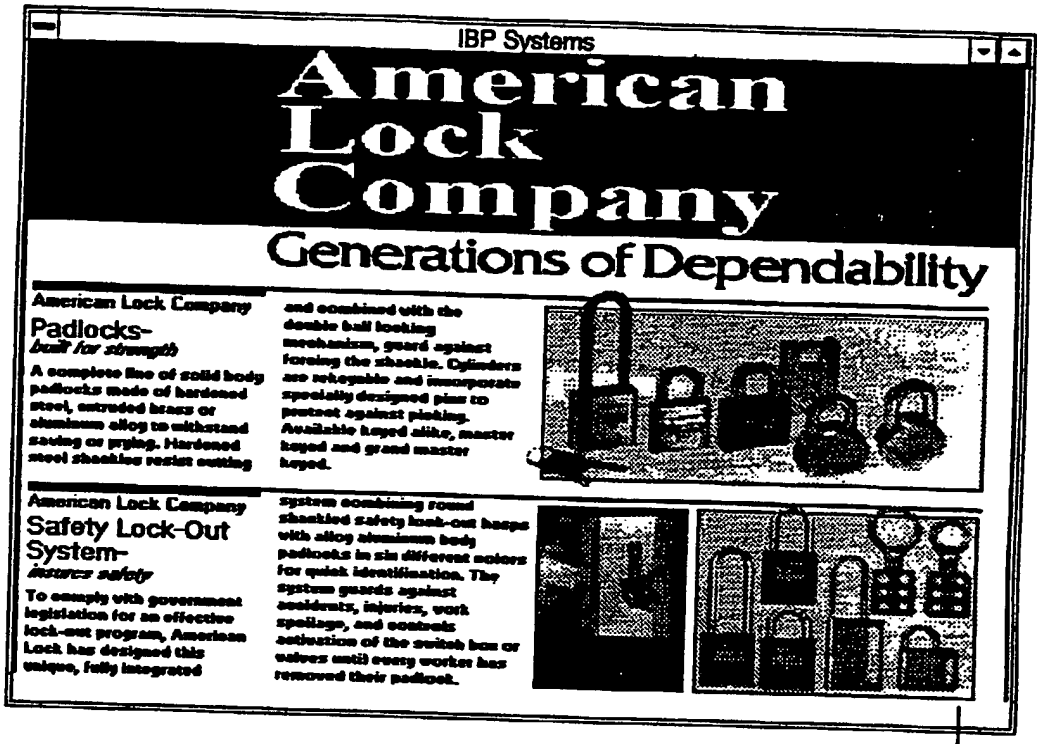


FIGURE 8(a)

280a

FIGURE 8(b)

280b

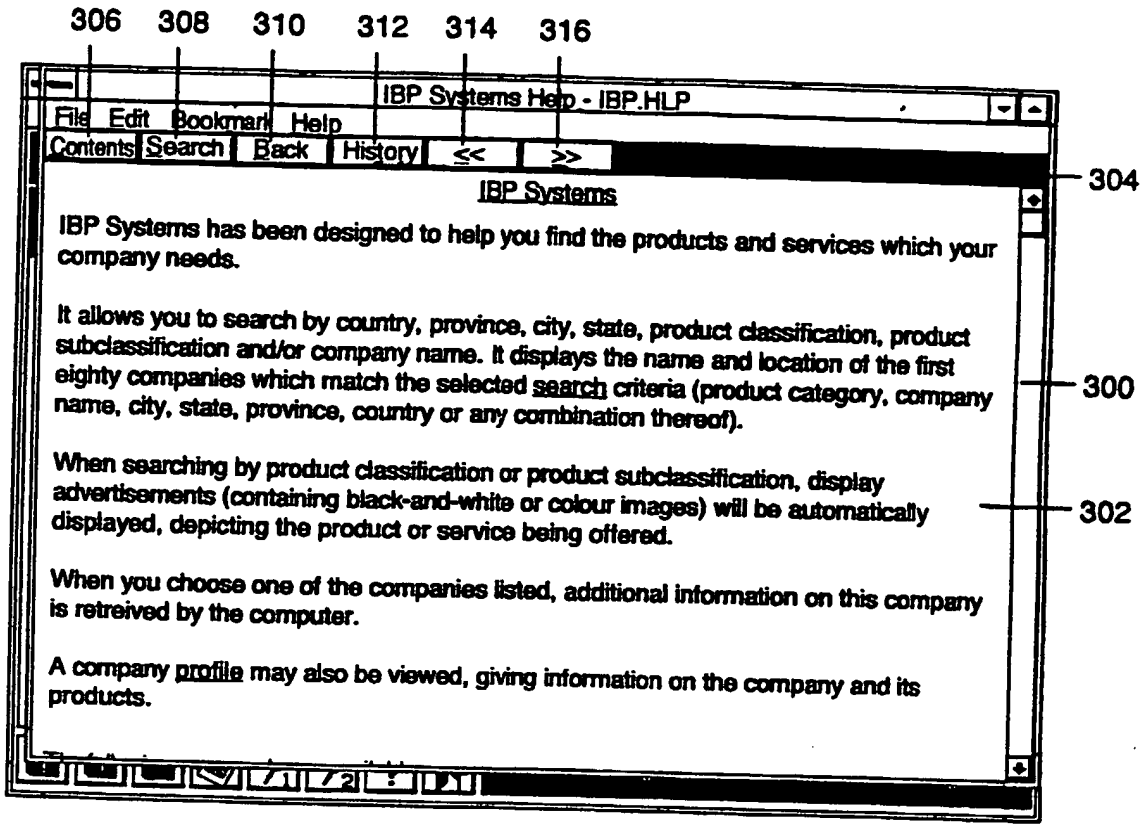

306
308
310
312
314
316
IBP Systems Help - IBP.HLP
File Edit Bookmark Help
Contents Search Back History << >>
304
IBP Systems
IBP Systems has been designed to help you find the products and services which your company needs.
It allows you to search by country, province, city, state, product classification, product subclassification and/or company name. It displays the name and location of the first eighty companies which match the selected search criteria (product category, company name, city, state, province, country or any combination thereof).
300
When searching by product classification or product subclassification, display advertisements (containing black-and-white or colour images) will be automatically displayed, depicting the product or service being offered.
302
When you choose one of the companies listed, additional information on this company is retreived by the computer.
A company profile may also be viewed, giving information on the company and its products.

FIGURE 9

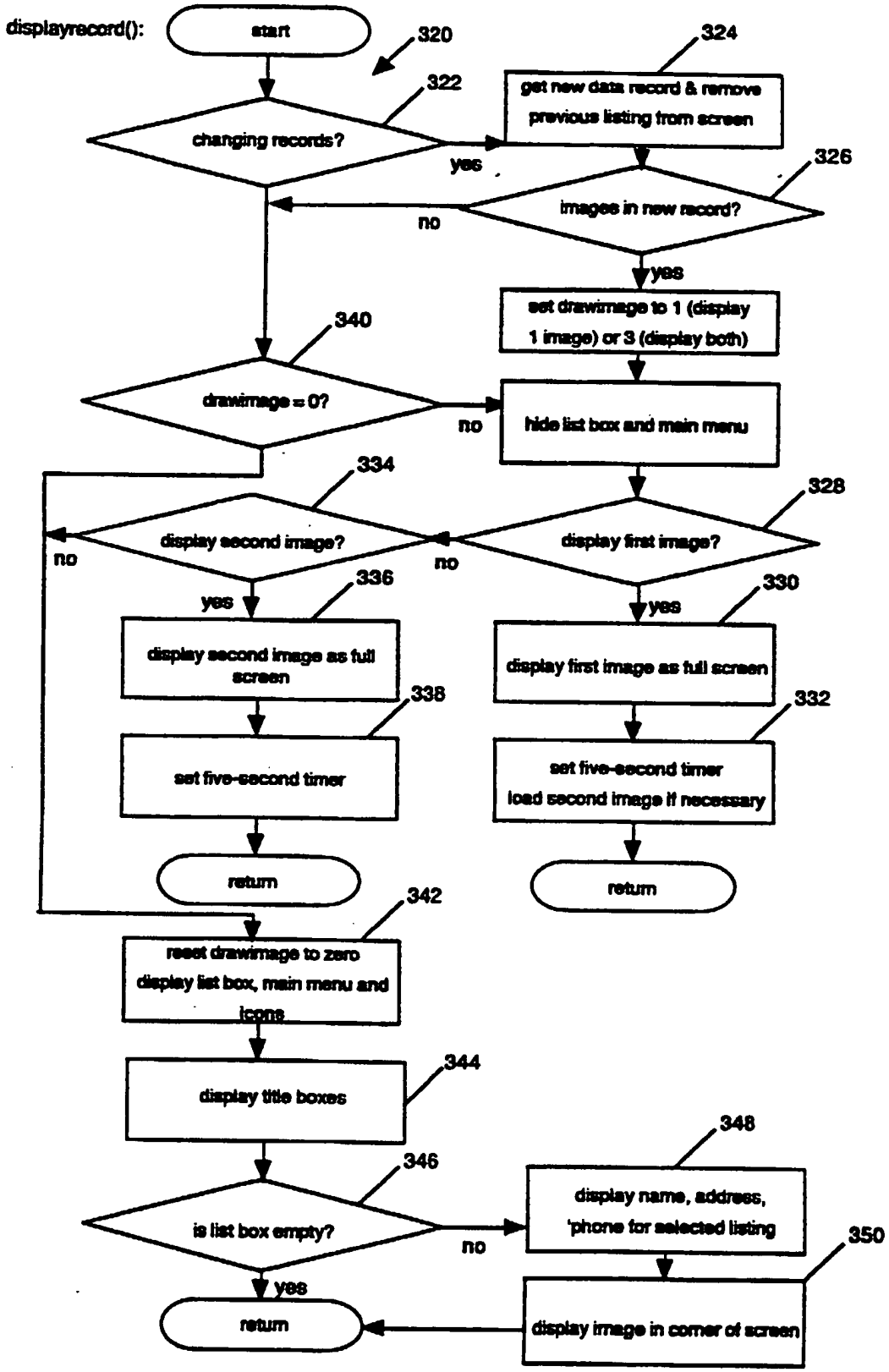

displayrecord():
start
320
changing records?
322
yes
get new data record & remove previous listing from screen
324
images in new record?
326
no
yes
set drawimage to 1 (display 1 image) or 3 (display both)
drawimage = 0?
340
no
hide list box and main menu
display second image?
334
no
display first image?
328
no
yes
display second image as full screen
336
yes
display first image as full screen
330
set five-second timer
338
set five-second timer load second image if necessary
332
return
return
reset drawimage to zero display list box, main menu and icons
342
display title boxes
344
is list box empty?
346
no
display name, address, 'phone for selected listing
348
display image in corner of screen
350
yes
return

FIGURE 10

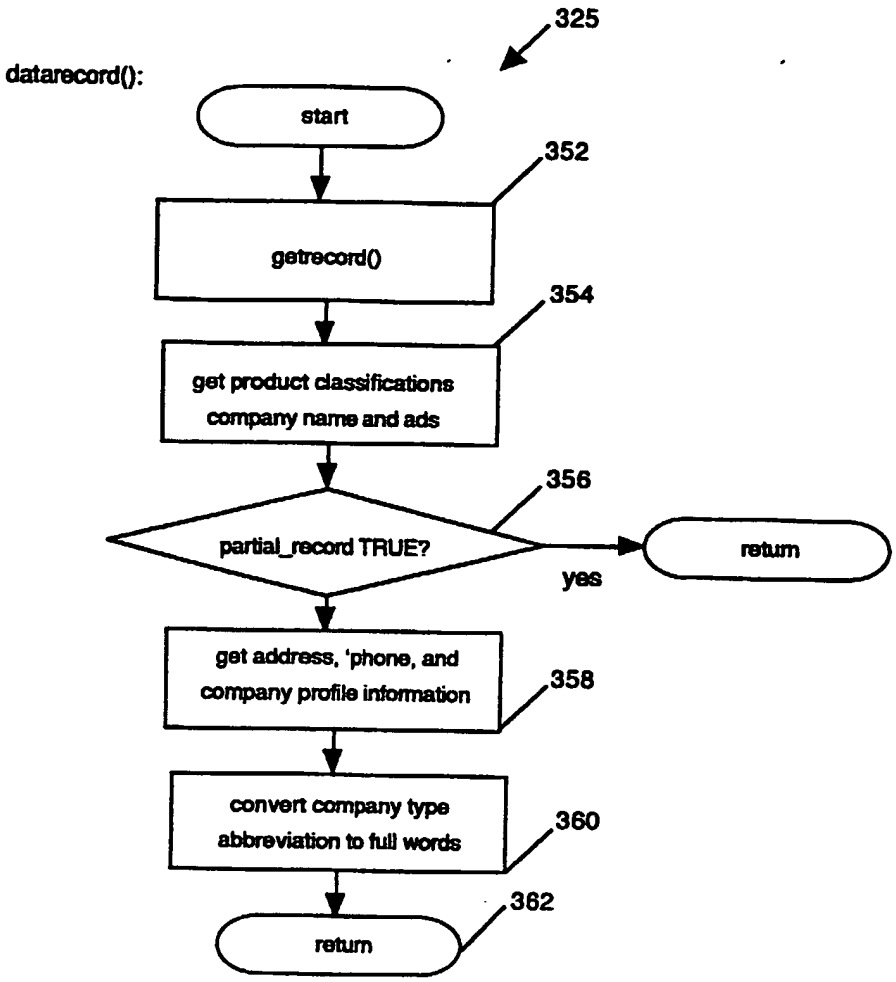

datarecord():
325
start
352
getrecord()
354
get product classifications
company name and ads
356
partial_record TRUE?
yes
return
get address, 'phone, and
company profile information
358
convert company type
abbreviation to full words
360
362
return

FIGURE 11

showimage():

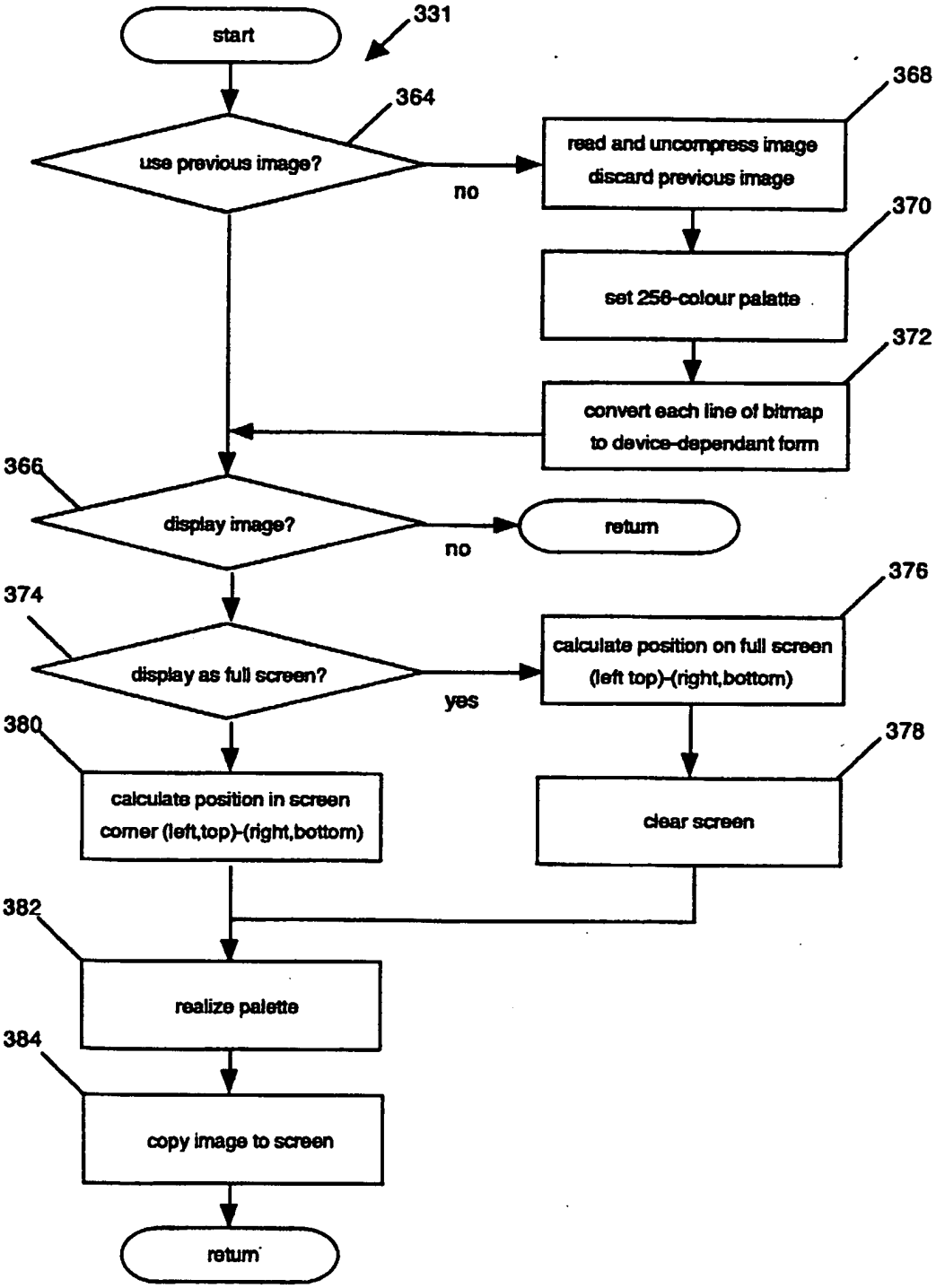


FIGURE 12

openimagefile():

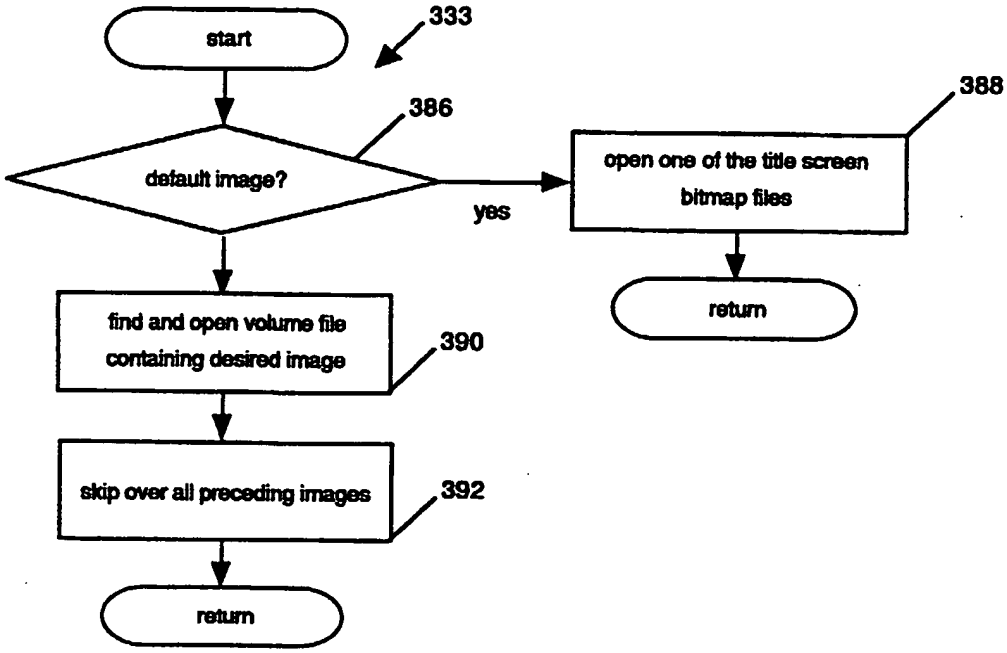


FIGURE 13

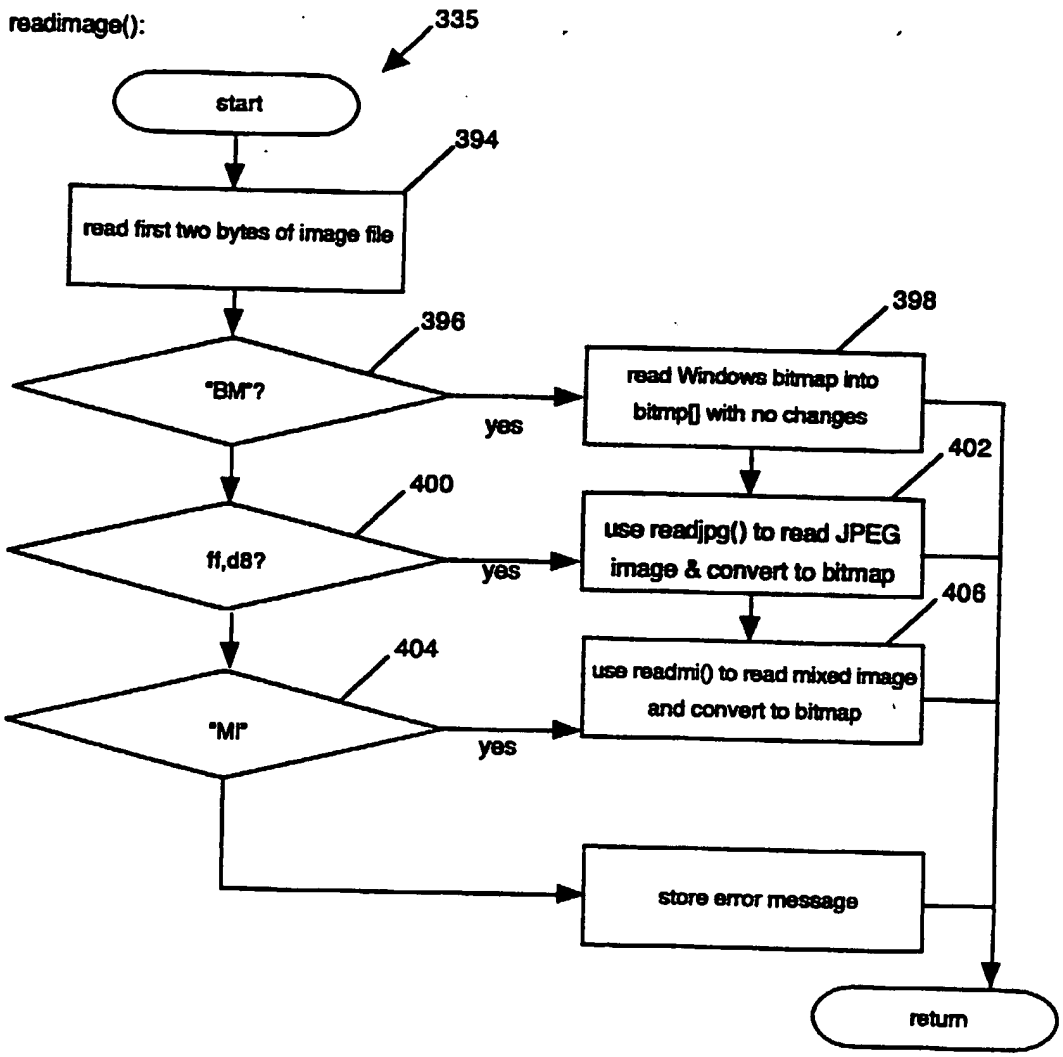

readimage():
335
start
394
read first two bytes of image file
396
"BM"?
yes
398
read Windows bitmap into bitmp[] with no changes
400
ff,d8?
yes
402
use readjpg() to read JPEG image & convert to bitmap
404
"MI"
yes
406
use readmi() to read mixed image and convert to bitmap
store error message
return

FIGURE 14

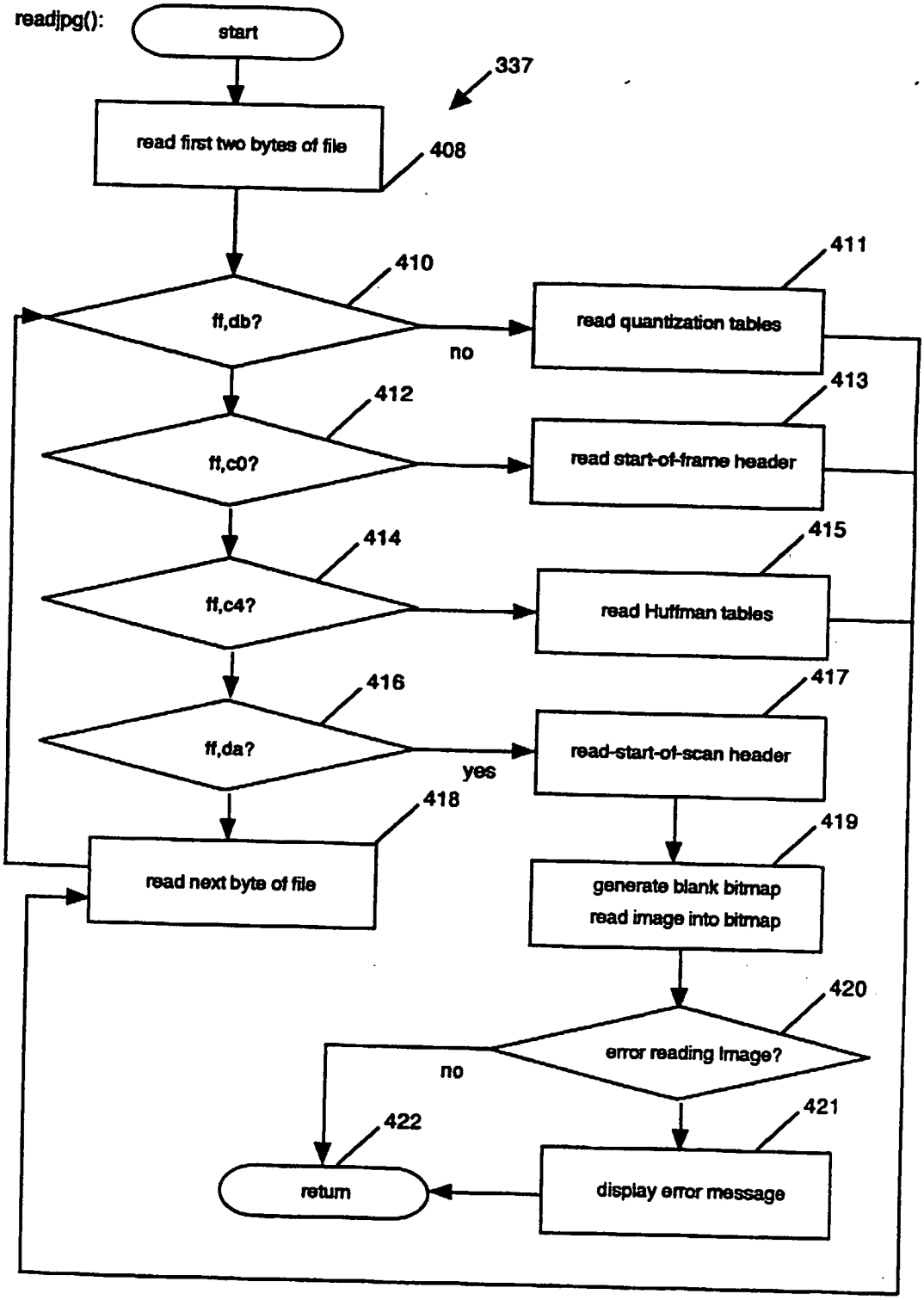

readjpg():
start
337
read first two bytes of file
408
410
ff,db?
no
411
read quantization tables
412
ff,c0?
413
read start-of-frame header
414
ff,c4?
415
read Huffman tables
416
ff,da?
yes
417
read-start-of-scan header
418
read next byte of file
419
generate blank bitmap
read image into bitmap
420
error reading image?
no
421
display error message
422
return

FIGURE 15

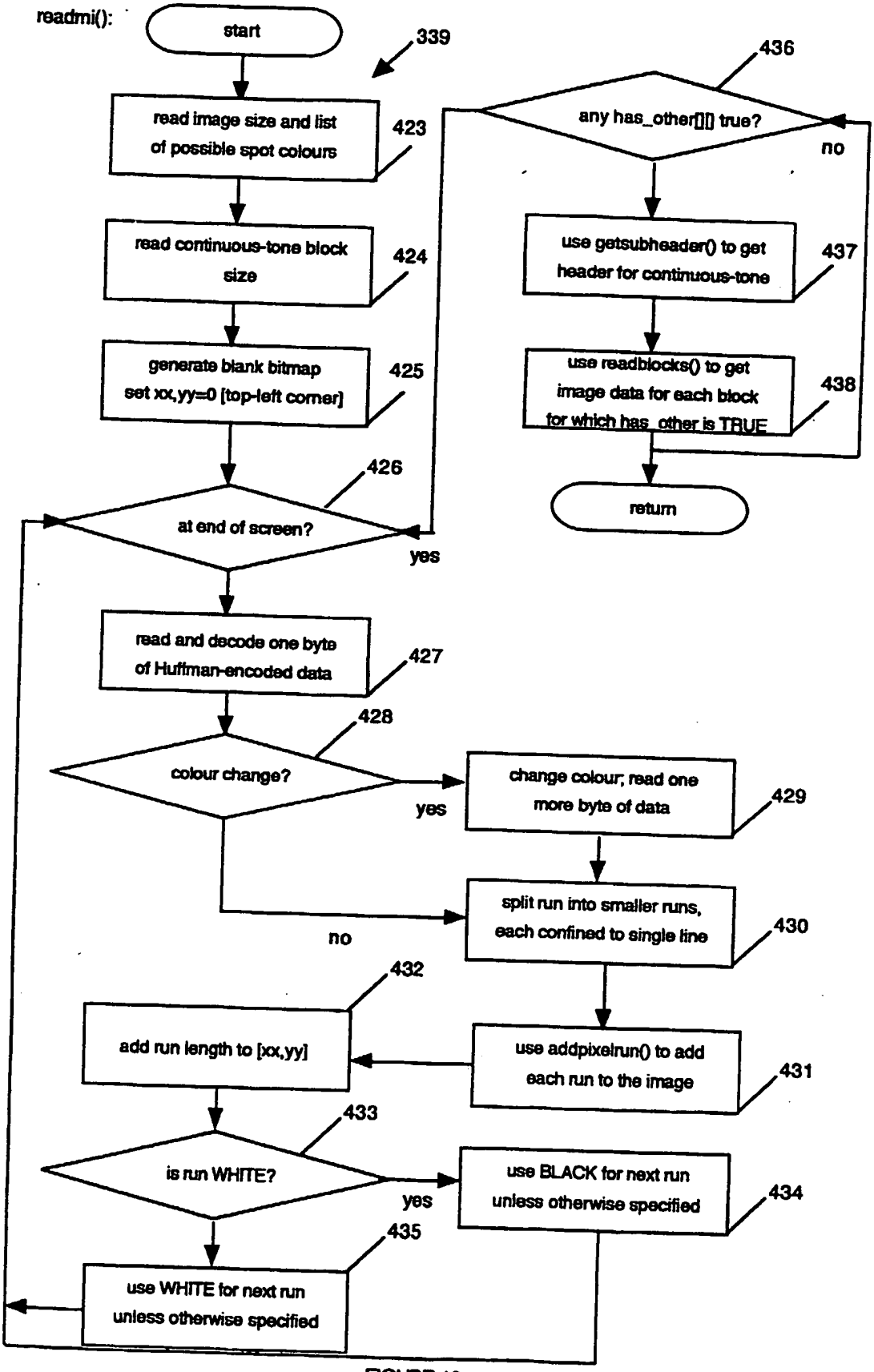

readmi():
start
339
read image size and list of possible spot colours
423
read continuous-tone block size
424
generate blank bitmap set xx,yy=0 [top-left corner]
425
426
at end of screen?
yes
read and decode one byte of Huffman-encoded data
427
428
colour change?
yes
change colour; read one more byte of data
429
no
split run into smaller runs, each confined to single line
430
432
add run length to [xx,yy]
use addpixelrun() to add each run to the image
431
433
is run WHITE?
yes
use BLACK for next run unless otherwise specified
434
435
use WHITE for next run unless otherwise specified
436
any has_other[][] true?
no
use getsubheader() to get header for continuous-tone
437
use readblocks() to get image data for each block for which has_other is TRUE
438
return

FIGURE 16

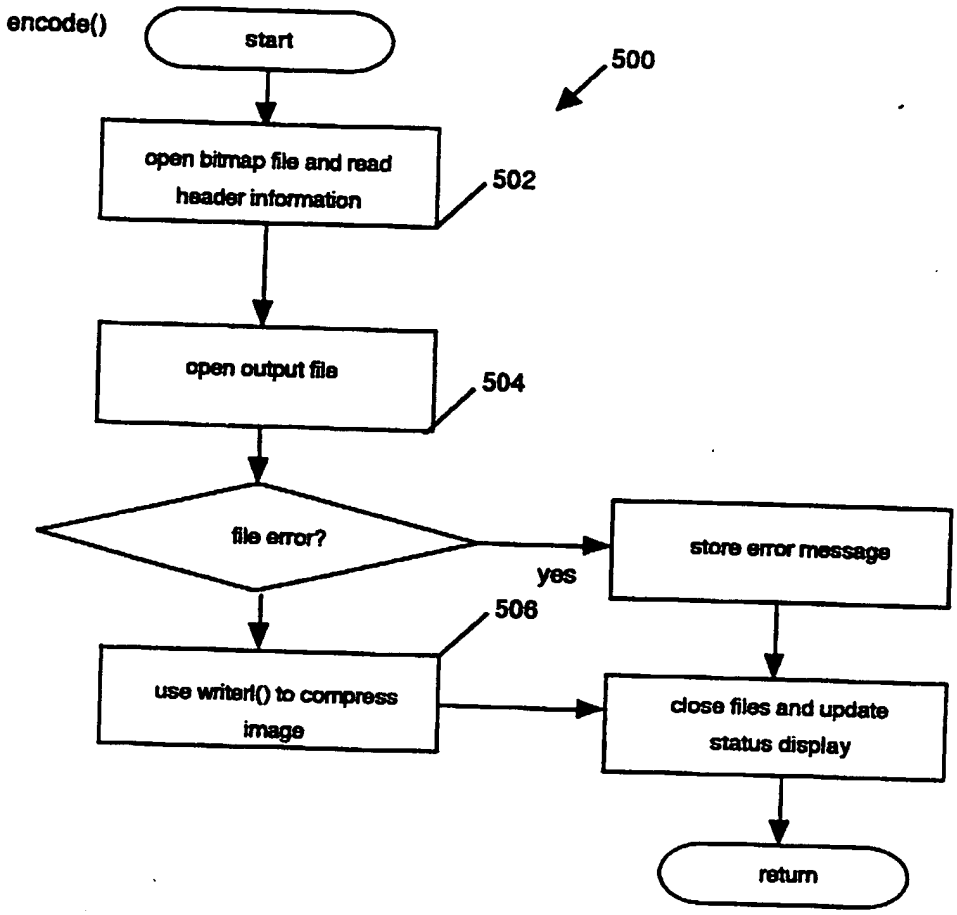

encode()
start
500
open bitmap file and read header information
502
open output file
504
file error?
yes
store error message
506
use writerl() to compress image
close files and update status display
return

FIGURE 17

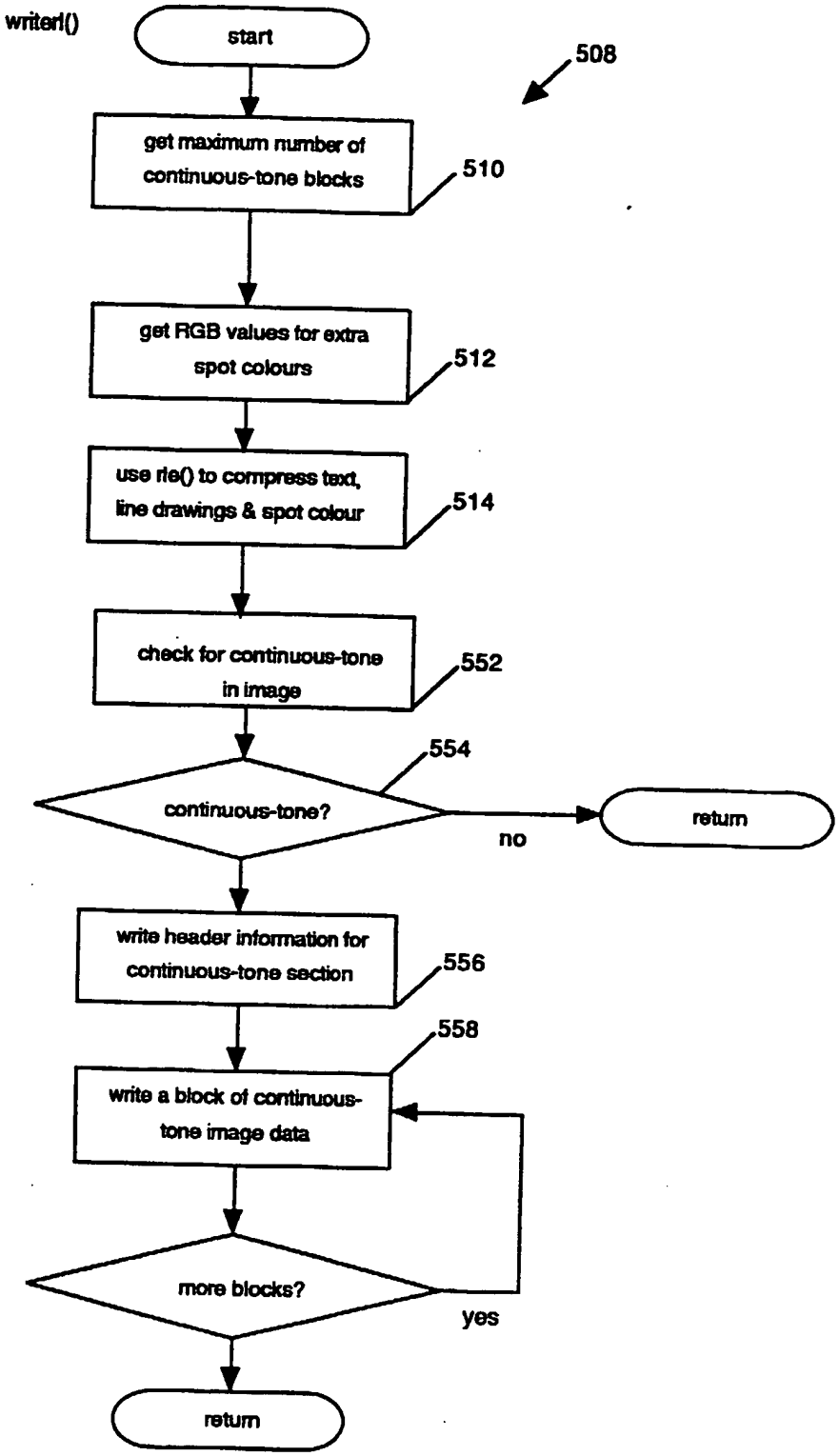

writerl()
start
508
get maximum number of continuous-tone blocks
510
get RGB values for extra spot colours
512
use rle() to compress text, line drawings & spot colour
514
check for continuous-tone in image
552
554
continuous-tone?
no
return
write header information for continuous-tone section
556
558
write a block of continuous-tone image data
more blocks?
yes
return

FIGURE 18

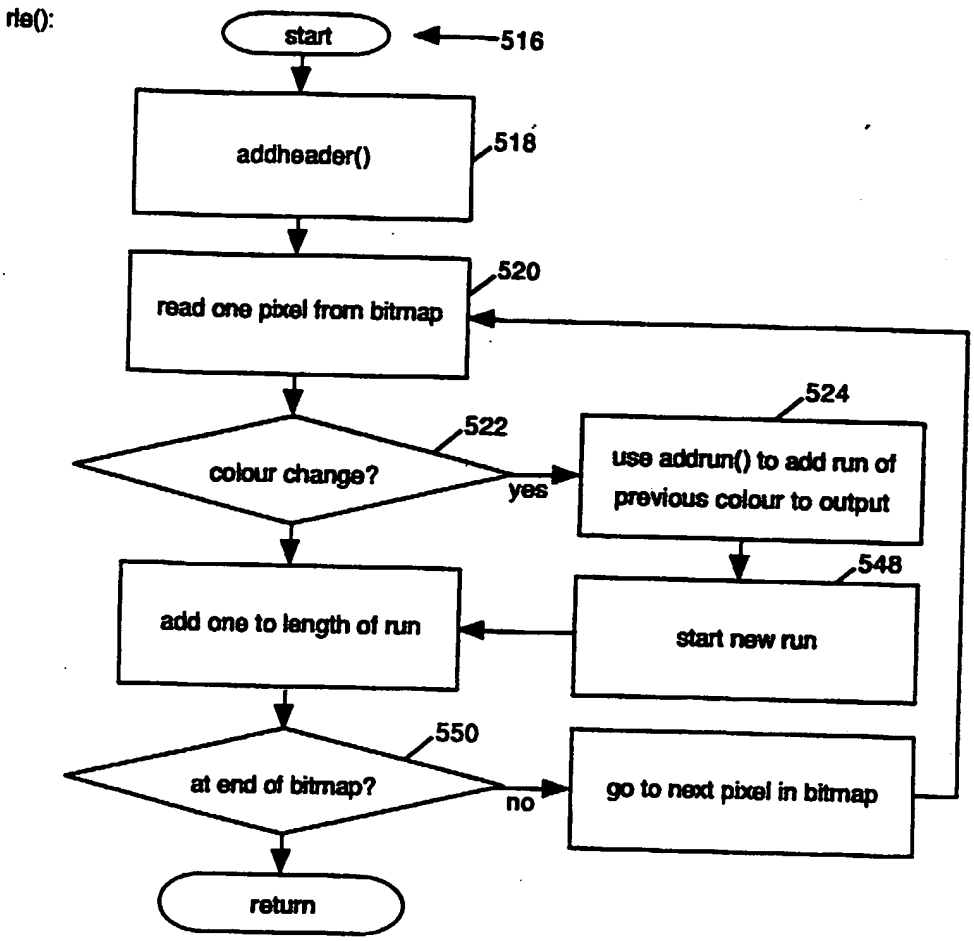

rle():
start
516
addheader()
518
read one pixel from bitmap
520
colour change?
522
yes
use addrun() to add run of previous colour to output
524
add one to length of run
start new run
548
at end of bitmap?
550
no
go to next pixel in bitmap
return

FIGURE 19

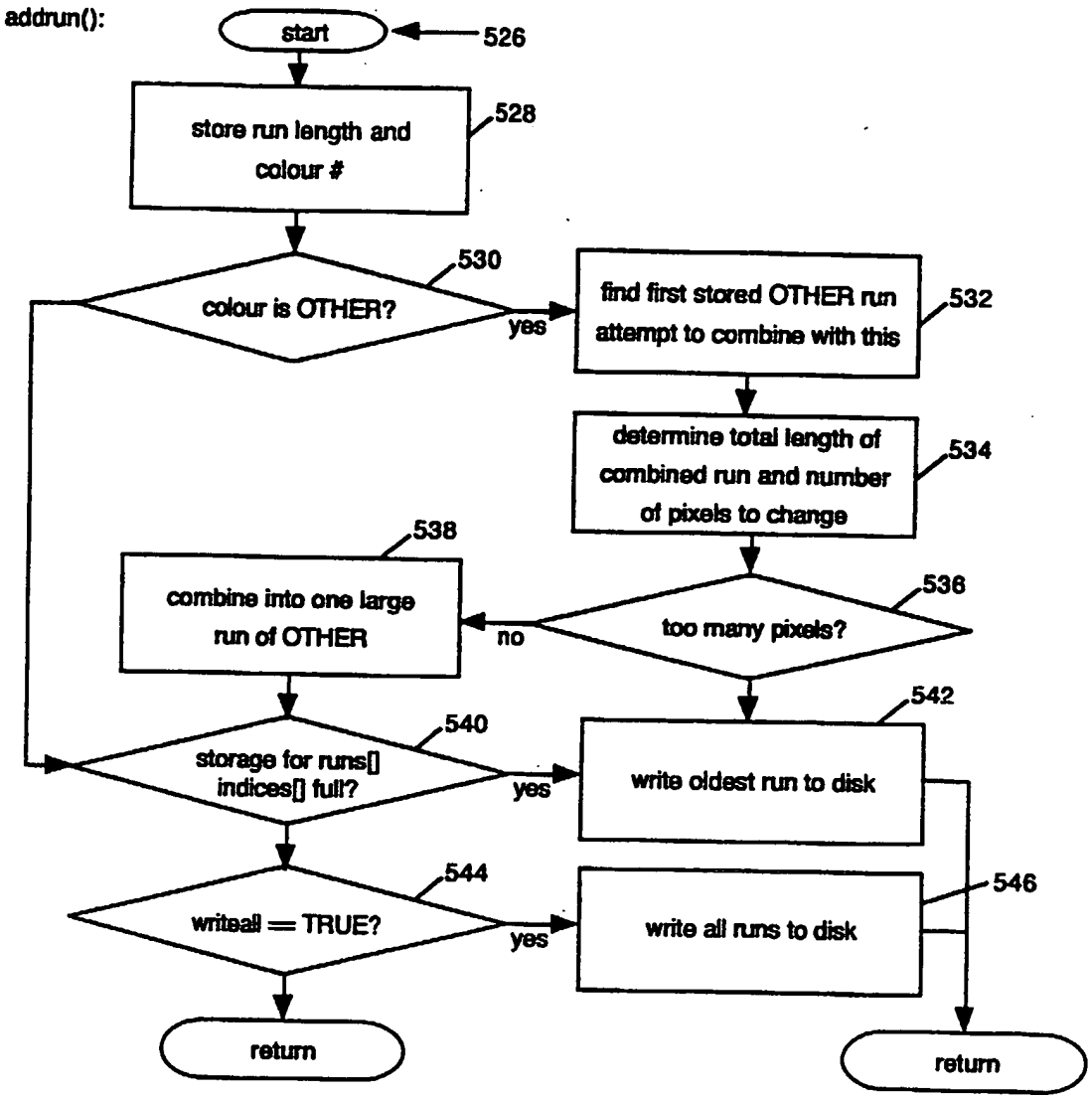

addrun():
start
526
store run length and colour #
528
colour is OTHER?
530
yes
find first stored OTHER run attempt to combine with this
532
determine total length of combined run and number of pixels to change
534
too many pixels?
536
no
combine into one large run of OTHER
538
storage for runs[] indices[] full?
540
yes
write oldest run to disk
542
writeall = TRUE?
544
yes
write all runs to disk
546
return
return

FIGURE 20

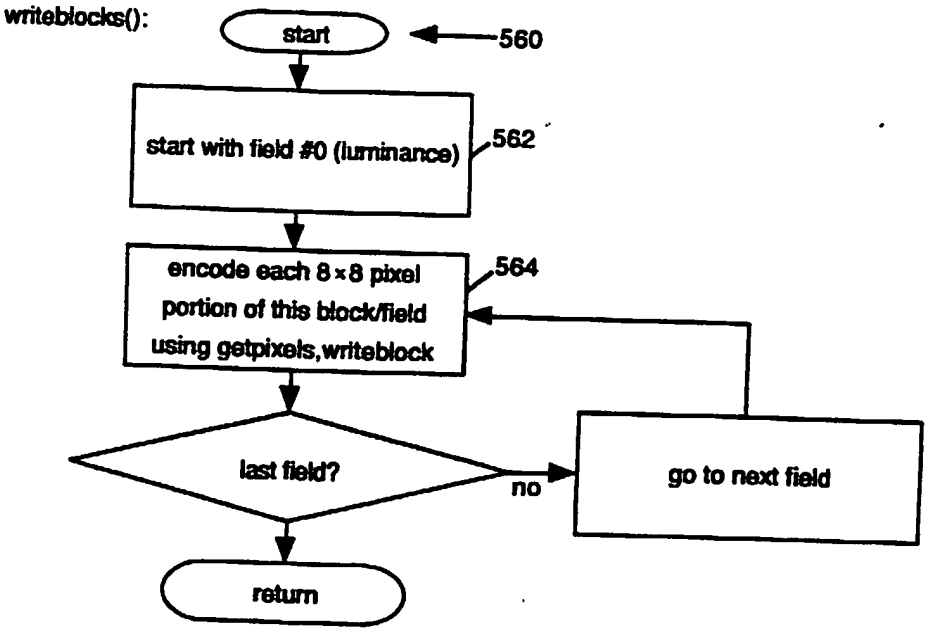

writeblocks():
start
560
start with field #0 (luminance)
562
encode each 8×8 pixel
portion of this block/field
using getpixels,writeblock
564
last field?
no
go to next field
return

FIGURE 21

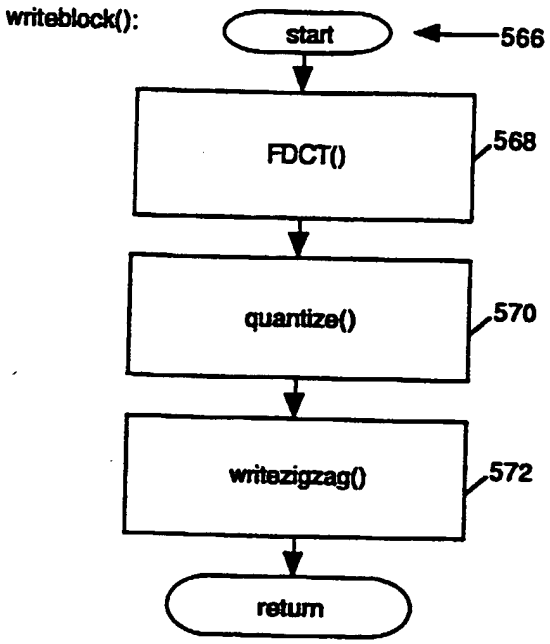

writeblock():
start
566
FDCT()
568
quantize()
570
writezigzag()
572
return

FIGURE 22

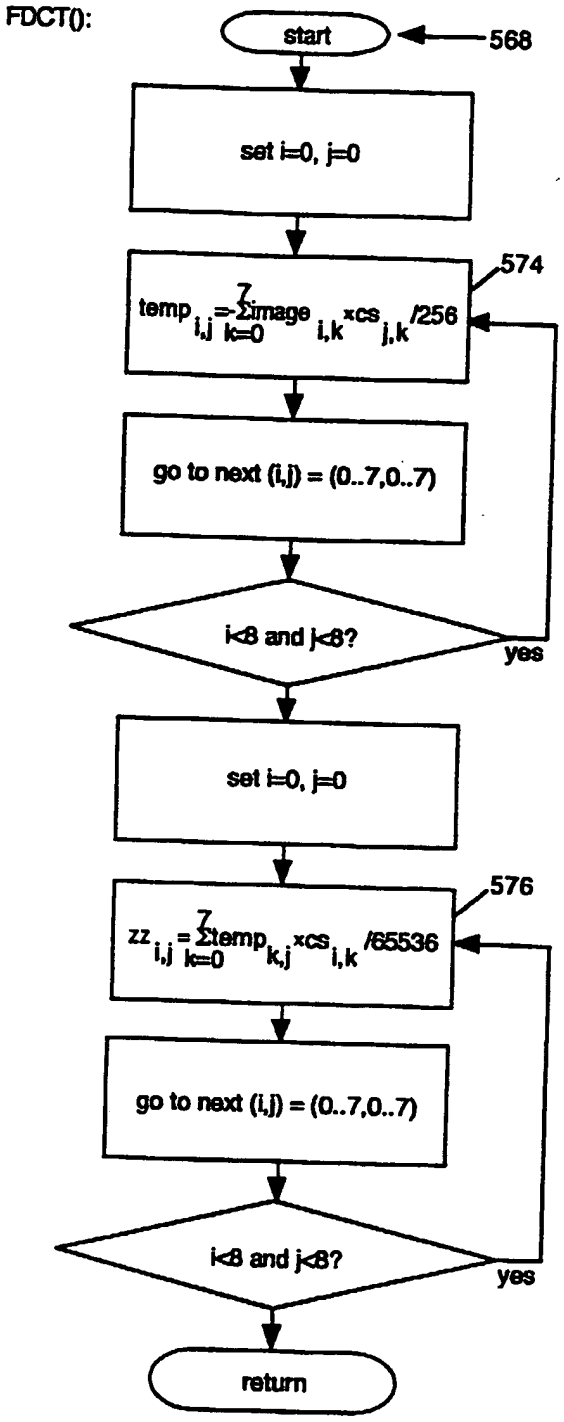

FDCT():
start
568
set i=0, j=0
574
$temp_{i,j} = \sum_{k=0}^{7} image_{i,k} \times cs_{j,k} / 256$
go to next (i,j) = (0..7,0..7)
i<8 and j<8?
yes
set i=0, j=0
576
$zz_{i,j} = \sum_{k=0}^{7} temp_{k,j} \times cs_{i,k} / 65536$
go to next (i,j) = (0..7,0..7)
i<8 and j<8?
yes
return

FIGURE 23

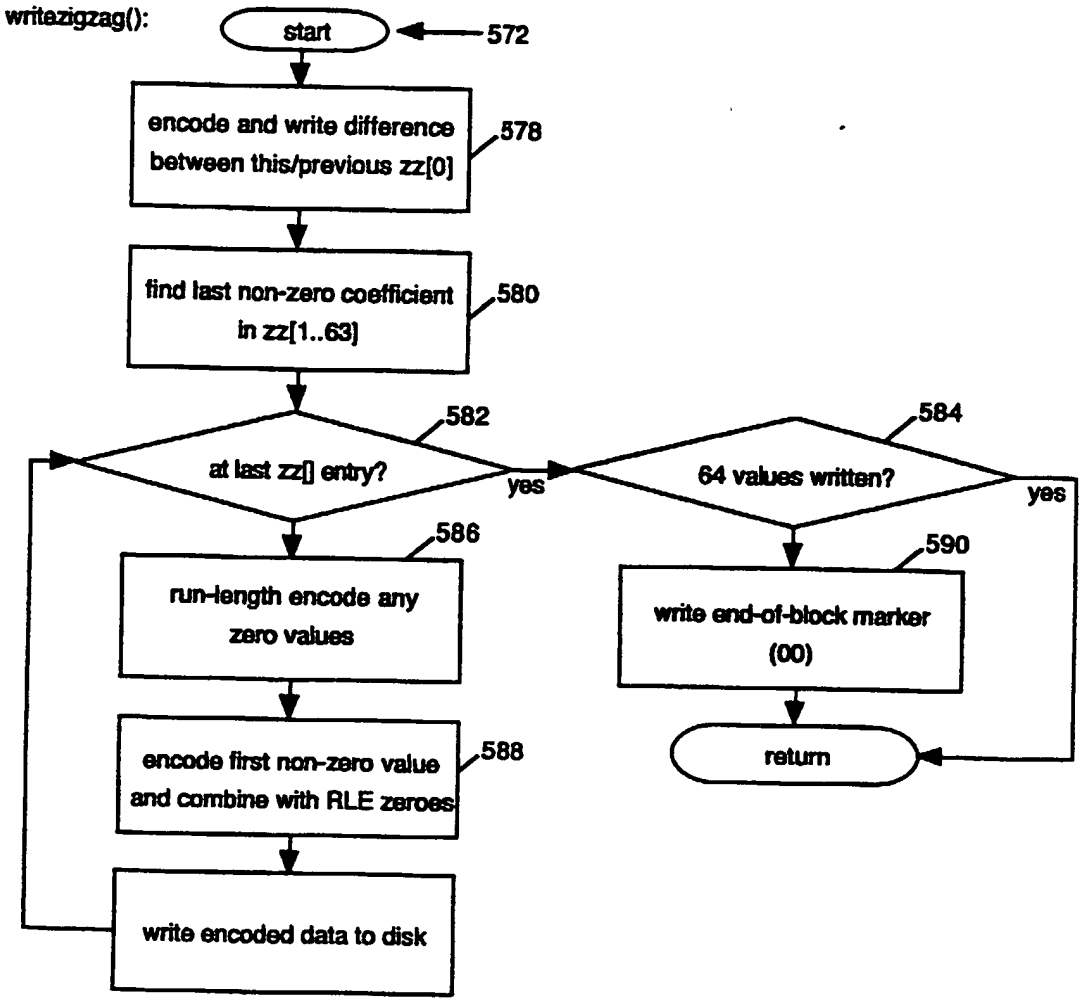

writezigzag():
start
572
encode and write difference between this/previous zz[0]
578
find last non-zero coefficient in zz[1..63]
580
at last zz[] entry?
582
yes
64 values written?
584
yes
run-length encode any zero values
586
encode first non-zero value and combine with RLE zeroes
588
write encoded data to disk
write end-of-block marker (00)
590
return

FIGURE 24

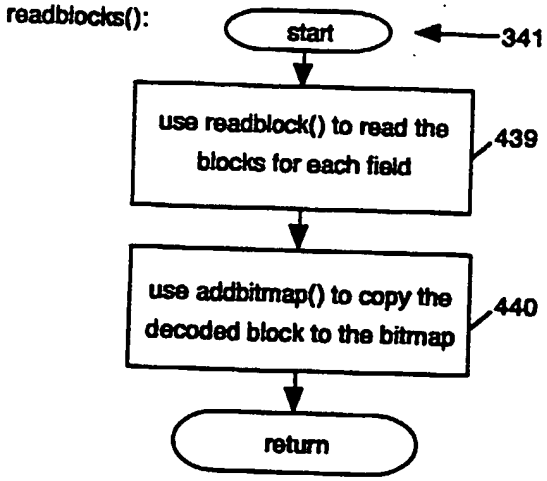

readblocks():
start
341
use readblock() to read the blocks for each field
439
use addbitmap() to copy the decoded block to the bitmap
440
return

FIGURE 25(a)

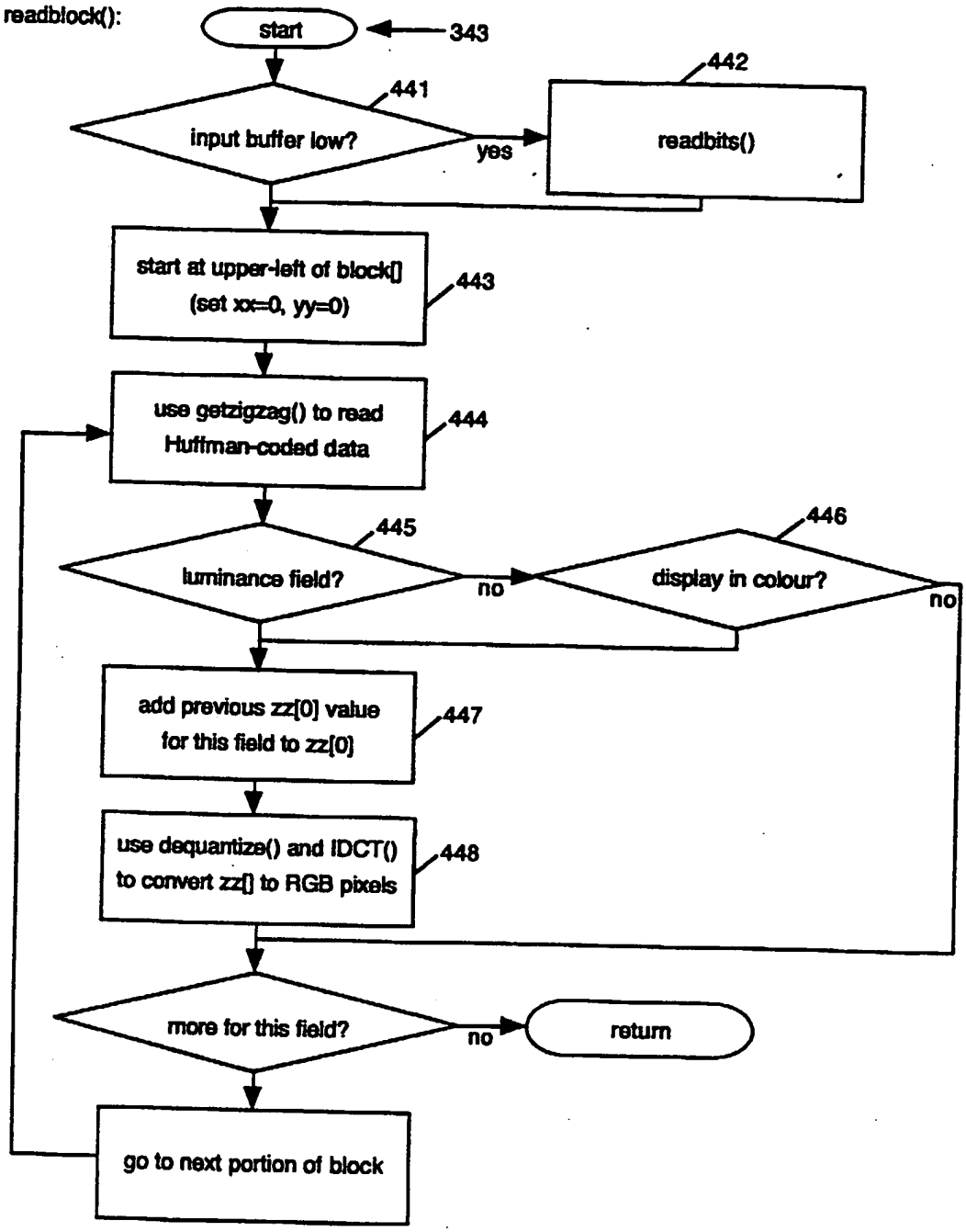


FIGURE 25(b)

SUBSTITUTE SHEET

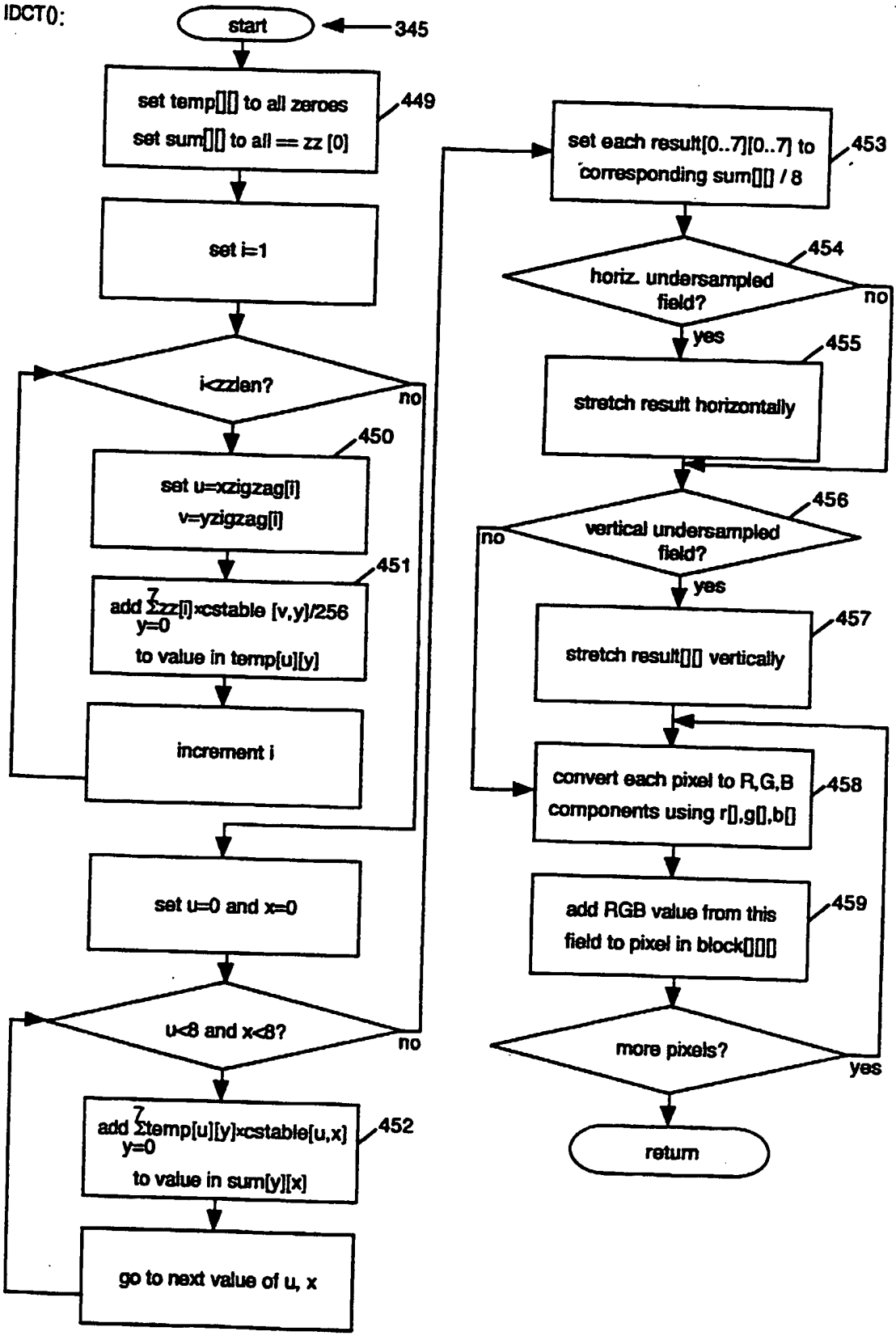


FIGURE 26

SUBSTITUTE SHEET

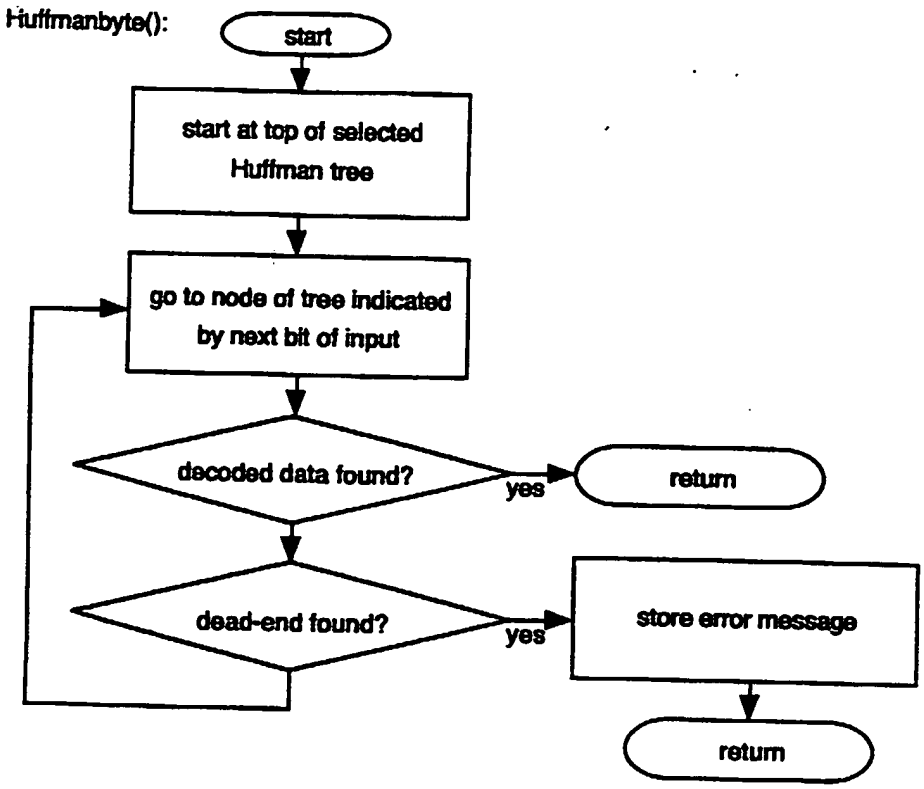

Huffmanbyte():
start
start at top of selected Huffman tree
go to node of tree indicated by next bit of input
decoded data found?
yes
return
dead-end found?
yes
store error message
return

FIGURE 27

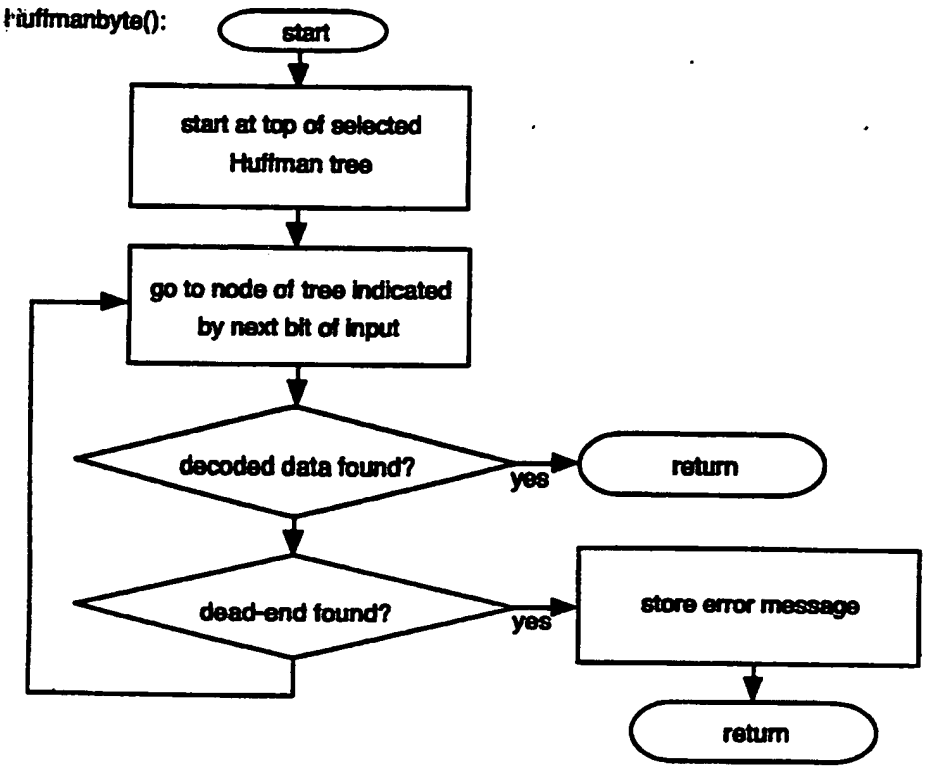

Huffmanbyte():
start
start at top of selected Huffman tree
go to node of tree indicated by next bit of input
decoded data found?
yes
return
dead-end found?
yes
store error message
return

FIGURE 27